Universidad Zaragoza

TRABAJO FIN DE GRADO

ESTUDIO DE SENSORES iBEACONS Y SU INTEGRACION PARA UN SISTEMA DE MONITORIZACION NO INTRUSIVA PARA PERSONAS DE LA TERCERA EDAD

Autor

José Ángel Caudevilla Casaús

Director

Rafael Ferrer Sánchez

Ponente

Enrique Torres Moreno

Grado en ingeniería informática

Centro politécnico superior de la Universidad de Zaragoza

2018

TRABAJOS DE FIN DE GRADO / FIN DE MÁSTER

Escuela de Ingeniería y Arquitectura Universidad Zaragoza

DECLARACIÓN DE AUTORÍA Y ORIGINALIDAD

(Este documento debe acompañar al Trabajo Fin de Grado (TFG)/Trabajo Fin de Máster (TFM) cuando sea depositado para su evaluación).

D./Dª. José Ángel Caudevilla Casaús,

con nº de DNI 18063992E en aplicación de lo dispuesto en el art. 14 (Derechos de autor) del Acuerdo de 11 de septiembre de 2014, del Consejo de Gobierno, por el que se aprueba el Reglamento de los TFG y TFM de la Universidad de Zaragoza,

Declaro que el presente Trabajo de Fin de (Grado/Máster) Grado, (Título del Trabajo) Estudio de sensores iBeacons y su integración para un sistema de monitorización no intrusiva para personas de la tercera edad.

es de mi autoría y es original, no habiéndose utilizado fuente sin ser citada debidamente.

Zaragoza, 17 de Septiembre de 2018

Fdo: José Ángel Caudevilla Casaús

Agradecimientos

Quiero agradecer a mi madre por darme la oportunidad que me ha dado y que ella no pudo tener, a mi padre y a mi amigo Eric por escucharme y ayudarme en los momentos difíciles.

También quiero agradecer a mis compañeros y amigos de la universidad, por aprender de ellos y compartir este camino juntos, sacándome una sonrisa y ayudándome cuando he tenido dificultades.

Agradecer a mis compañeros de clase con los que he tenido el placer de colaborar durante las prácticas y las asignaturas de la carrera.

Al resto de mis amigos por estar ahí cuando los he necesitado.

Quiero agradecer también a la empresa Neki, y en especial a Rafa, la oportunidad que me han dado para formar parte de este interesante y a la vez bonito proyecto.

Y, por último, y no menos importante, agradecer la labor de mi tutor Enrique por enseñarme y guiarme a lo largo de este trabajo.

Gracias a todos.

Resumen

El Internet de las Cosas (IoT) camina imparable para que vivamos conectados en todo momento. Gracias a él se puede tener casi cualquier tipo de dispositivos electrónicos conectados a internet.

Neki es una empresa que ofrece un servicio de localización GPS para personas que necesitan cierta supervisión. Como es el caso de niños y personas mayores con riesgo a desorientarse.

A través de una aplicación móvil, el cuidador puede saber cuándo la persona al cuidado se está alejando.

El servicio permite controlar a las personas cuando están fuera de casa, pero no permite controlarlas dentro de casa, en el caso de las personas mayores mayoritariamente validas, pueden necesitar cierta supervisión para realizar algunas tareas.

Estas personas tienen riesgo de sufrir periodos de desorientación o tener un percance que requiera ayuda. Debido a motivos económicos o por falta de tiempo, nadie puede supervisar a estas personas o ellas mismas son las que prefieren vivir solas.

El objetivo de este proyecto es desarrollar un sistema de monitorización no intrusivo para personas de la tercera edad.

Este sistema usa unas balizas denominadas Beacons para monitorizar la vivienda, detectando la actividad de una vivienda invadiendo mínimamente la intimidad de la persona. A través de internet se va a enviar esta información a la plataforma de Neki en donde se determina si todo va bien o mal.

La plataforma de Neki llevar a cabo toda la lógica de la aplicación y la interacción con los usuarios (las personas encargadas del cuidado), recibiendo notificaciones en el caso de que algo vaya mal en la vivienda.

Para desarrollar esta aplicación se han evaluado los Beacons, comprobando que son capaces de detectar las actividades de una vivienda. A partir de esta evaluación se ha desarrollado el sistema descrito anteriormente.

Índice

1. Introducción

Neki, una startup situado en Zaragoza, ofrece un servicio para el seguimiento de personas a través de localizadores GPS. Mediante una aplicación móvil se puede saber dónde se encuentra una persona en todo momento o si necesita ayuda.

El uso habitual es en personas mayores, mayoritariamente validas, pero que pueden sufrir episodios de desorientación al salir de casa. El principal objetivo es que estos ancianos semidependientes puedan hacer una vida normal, pero si se pierden o desorientan tengan un modo fácil de avisar al cuidador.

Gracias a la localización por GPS se pueden establecer unas zonas de seguridad. Si el usuario sale de la zona establecida se producirá una alarma. El cuidador a través de la aplicación en el teléfono móvil podrá decidir qué hacer, por ejemplo, llamarlo por teléfono o ir a buscarlo o si es una situación normal cancelar la alarma.

Además, el dispositivo tiene un botón de pánico o de SOS. El usuario puede pulsarlo en cualquier momento que sienta la necesidad. El cuidador podrá atenderlo vía telefónica, con la ventaja añadida de tenerlo localizado.

Lo mismo es aplicable a niños, adolescentes u otras personas que requieran ese servicio.

Con este servicio de localización se puede controlar si una persona sale de casa y necesita ayuda, pero no se puede controlar lo que está ocurriendo dentro de la vivienda. Por ejemplo, puede darse el caso de que la persona no pueda levantarse de su cama y necesitar ayuda, que no se haya tomado sus pastillas o se haya dejado la puerta de casa abierta.

Con esto en mente, desde la empresa Neki se busca desarrollar un sistema de monitorización en viviendas de manera no intrusiva. Centrándose en personas pertenecientes a la tercera edad y que son semidependientes.

Dentro de este grupo se encuentran personas que pueden valerse por sí mismas, pero necesitan supervisión. Por ejemplo, personas con riesgo de desorientarse o enfermos de Alzheimer no avanzado.

Este sistema se caracteriza en que no busca invadir la intimidad del monitorizado. Esta característica se va a llevar a cabo colocando una serie de sensores estratégicamente, de manera que estos sensores sean capaces de detectar las actividades cotidianas de una vivienda.

Se quiere ofrecer un servicio a los usuarios encargados del cuidado notificándolos cuando algo no vaya bien en la vivienda. Para ofrecer este servicio el sistema va a estar conectado a internet, gracias a la aparición del *Internet of Things* se pueden tener casi cualquier tipo de dispositivo electrónico conectado a internet.

El concepto *Internet of Things (IoT)* se refiere a la interconexión digital de objetos cotidianos con Internet o nuestro dispositivo móvil mediante Apps, y dotarlos de inteligencia para que automaticen tareas.

Mediante el uso de estos dispositivos basados en IoT se puede tener monitorizada a una persona de forma no intrusiva. Sabiendo si en un día no se ha abierto la nevera o si no se ha encendido la tele en todo el día, notificando a través de internet estos eventos.

El sistema se caracteriza en el uso de una serie de reglas de inferencia establecidas por la persona encargada del cuidado. Estas reglas de inferencia corresponden a actividades diarias de la persona semidependiente, como por ejemplo que no se haya abierto la nevera en todo el día.

Si una de estas reglas de inferencia no se cumple la persona encargada del cuidado recibirá a través de la APP de Neki una notificación y podrá decidir qué hacer. Por ejemplo, llamar a la persona por teléfono parar comprobar que todo va bien.

Para poder detectar las actividades de una vivienda se van a usar sensores ya certificados con marcado CE. Este marcado garantiza que el producto cumple con la legislación de la unión europea y se permite su libre circulación.

En este trabajo final de grado se busca estudiar si un sensor de tipo acelerómetro proporcionado por la empresa Neki es capaz de detectar los movimientos producidos en las puertas de una vivienda.

Existe una serie de alternativas para la detección de movimientos, como puede ser el uso de sensores mecánicos, sensores de luz, sensores magnéticos…

Neki está interesado en la versión acelerómetro porque estos dispositivos de pequeño tamaño se pueden instalar en casi cualquier lugar sin ningún cableado. Únicamente necesitan ser pegados a una puerta para sean capaces de detectar su movimiento.

Al ser un sistema de tipo IoT se requiere conexión a internet para su uso. Muchas personas mayores no suelen tener acceso a internet en sus viviendas. Por el contrario, aquellas que si poseen están sujetas a la configuración de red de su proveedor de internet.

Como no se quiere depender de configuraciones de proveedores dispares el sistema se va a conectar a internet a través de una red móvil contratada por Neki.

Esta red móvil tiene una tarifa por consumo de datos. Lo que implica que este coste va a variar según el tráfico producido por la aplicación. Es interesante para Neki saber cuánto tráfico se está generando para estimar los costes de este servicio.

Otro objetivo de este TFG es estimar el tráfico de datos generado por el sistema.

Se ha desarrollar una pasarela software o middleware encargada de recoger la información de los sensores y enviar las actividades generadas por estos sensores a la plataforma de Neki. Estas actividades van a corresponder a movimientos de puerta.

En la plataforma de Neki se va a llevar a cabo la lógica del servicio. A su vez, esta plataforma se encarga de la comunicación con los tutores. A través de esta plataforma los tutores podrán establecer sus propias reglas de inferencia y recibirán las notificaciones cuando se detecte una inactividad en base a las reglas predefinidas.

Una vez implementado el middleware se ha verificado su funcionalidad para posteriormente integrarlo en la plataforma de Neki. Durante esta integración se realizarán las modificaciones necesarias en el middleware.

Una vez realizada la integración se ha documentado este trabajo final de grado.

2. Arquitectura

En esta sección se va a mostrar la arquitectura general del sistema. Definiendo claramente los tipos de usuarios y el rol que desempeñan.

Posteriormente se va describir cada uno de los componentes, sus características, tecnologías y su papel dentro del sistema.

a. Usuarios

Cada usuario va a solicitar que el sistema cumpla una serie de requisitos y atienda a unas necesidades propias del usuario. Durante el análisis se han identificado los siguientes tipos de usuarios:

- **Neki:** Es la empresa que quiere llevar a cabo el desarrollo del sistema de monitorización. Este usuario es el encargado de establecer los requisitos que tienen que cumplirse. Además, es deseable que la integración con su plataforma genere el menor número de problemas posible. Por último, va a esperar poder configurar los diferentes aspectos del sistema de una manera sencilla.
- **Monitorizado**: Dentro de este rol se encuentran las personas mayores semidependientes. Este usuario no va establecer ningún requisito, pero sí que va a esperar que el sistema no sea intrusivo con su intimidad.
- **Tutor**: En este sector se encuentran todos los hijos, familiares y tutores de los monitorizados. Por un lado, este usuario va a esperar que se cumplan unos requisitos. Por ejemplo, poder establecer las reglas de inferencia por el mismo. Por otro lado, va a espera que se cumplan unas necesidades relacionadas con el monitorizado. Por ejemplo, quiere recibir notificaciones cuando se detecte una inactividad.

b. Componentes

A continuación, se proporciona una visión general del sistema. Para cada uno de los compontes se va a realizar una breve explicación y sus características, junto con el papel que desempeña.

La siguiente imagen muestra la arquitectura general del sistema, en ella se muestran todos los componentes y usuarios del sistema, junto su descripción y el papel que desempeñan:

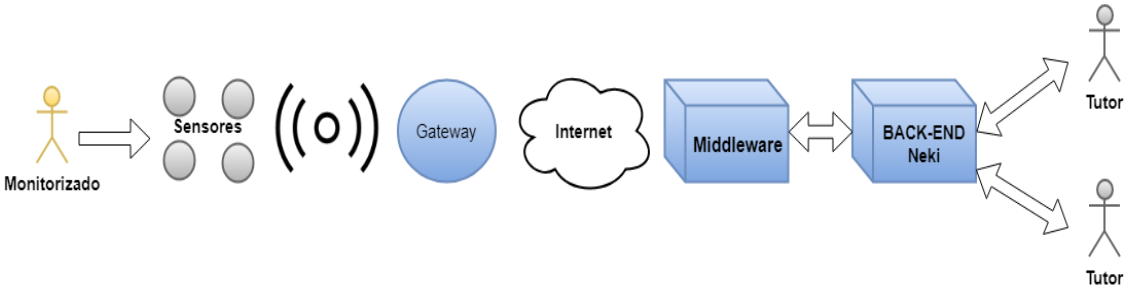

Figura 1. Arquitectura general

Sensores

Los sensores son dispositivos que se encargan de reunir información sobre el entorno físico. Están formados por células sensibles que detectan variaciones en una magnitud física y las convierten en señales eléctricas.

Gracias a sus características, estos dispositivos permiten conocer la temperatura, nivel de humedad, luminosidad, nivel de ruido y muchos otros datos de nuestra vivienda.

En el caso de nuestro sistema, el uso de sensores permite obtener información sobre lo que está ocurriendo en una vivienda. Conociendo si una persona ha encendido la tele, si ha ido al baño o si ha abierto la nevera.

Detectar estos eventos permite saber que la persona se encuentra bien.

Para desarrollar este sistema se han usado unos sensores denominados Beacons. Estos sensores son dispositivos inalámbricos que difunden señales broadcast de corta distancia a través de tecnología Bluetooth 4.0 (llamado Bluetooth Low Energy –BLE-).

BLE está siendo utilizado en dispositivos para dar servicios de señalización y localización. Gracias a la baja tasa de transmisión de datos que presentan ofrecen una larga duración de batería.

Para que nos hagamos una idea, un sensor alimentado por una "pila de botón" y con soporte BLE puede estar encendido durante meses, o incluso, llegar al año de funcionamiento sin necesidad de reemplazar la batería.

Debido a sus características la tecnología BLE se convierte en una tecnología clave para el desarrollo de proyectos IoT.

En este proyecto, BLE permite que los Beacons sean autosuficientes, ofrezcan una larga duración y se puedan colocar en cualquier lugar de la vivienda sin la necesidad de utilizar ningún tipo de cableado.

Figura 2. Beacons[1]

[1] https://steemkr.com/technology/@kevincortes13/balizas-informativas-beacon

Los Beacons proporcionados tiene un certificado CE, con este certificado se informa a los usuarios y autoridades competentes del cumplimiento de las especificaciones técnicas de dicho producto por parte del fabricante.

El certificado garantiza que se cumple la funcionalidad indicada por el fabricante. Los Beacons tiene un firmware no modificable, su modificación implicaría tener que volver a certificar el producto.

Usar el firmware del fabricante implica poder configurar únicamente lo que permite el fabricante.

En este proyecto se van a usar Beacons con acelerómetro. Estos dispositivos son capaces de detectar la aceleración para cada uno de los ejes x, y, z. Estos valores equivalen a Fuerzas G (una G equivale a la fuerza gravitatoria de la tierra).

Para configurar los Beacons proporcionados el fabricante ofrece una aplicación Android que conecta el Smartphone con las balizas a través de Bluetooth. Esta aplicación permite configurar la potencia de emisión, la frecuencia de emisión y la posibilidad de usar un disparador.

Estos Beacon emiten tramas a la frecuencia a la cual están configurados, dentro de estas tramas se encuentran los datos generados por el acelerómetro. El fabricante permite usar un disparador o trigger que emita las tramas cuando sucede algún evento, como puede ser pulsar un botón o la detección de una aceleración.

La aplicación móvil permite únicamente activar/desactivar esta opción, no permite configurar ningún otro parámetro como la sensibilidad del disparador.

La activación del trigger genera menos trafico debido a que únicamente se envían las tramas cuando se detecta un evento, produciendo un menor consumo de batería y una mayor duración del Beacon.

Por el contrario, no usar el trigger trae consigo un mayor tráfico de datos y un mayor consumo de batería. En este caso el periodo de muestreo produce un compromiso entre la precisión del Beacon y su consumo de batería.

Un periodo de muestreo bajo va a ser capaz de detectar cualquier pequeña aceleración, por contra va a consumir más batería, siendo necesario comprar baterías.

Por el contrario, un periodo de muestreo alto va a ser capaz de detectar la aceleración en momentos puntuales y va a consumir menos batería. Ahorrando costes en baterías.

Para resolver estas incógnitas se va a evaluar los Beacons proporcionados. Este estudio va a determinar si usando un trigger se pueden detectar los movimientos de una puerta, en caso contrario se va a obtener un valor de muestreo adecuado.

Durante la duración de este proyecto (más de tres meses) no se ha producido un descenso en el nivel de batería. Por esta razón se deja fuera el estudio del consumo producido por los beacons.

Gateway

Este dispositivo actúa como una interfaz de conexión entre los Beacons e internet. Su funcionamiento se basa en recoger los datos emitidos por los Beacons mediante BLE y enviarlos a un servidor remoto a través de internet mediante Wifi, 3G o ethernet. Estos datos son enviados a internet través de formato JSON.

Este gateway tiene un certificado CE y posee un firmware creado por el fabricante que solo es accesible por el mismo. Si se quisiera modificar el firmware habría que volver a pasar el certificado.

Se ha comentado al fabricante sobre la creación de un firmware propio y se ha ofrecido a realizarlo el mismo a cambio de un acuerdo comercial con Neki. Este acuerdo implica realizar una compra masiva de estos dispositivos.

Actualmente no interesa este acuerdo así que se va a estar sujeto al firmware del fabricante.

El gateway posee las siguientes características: permite leer la señal de varios Beacons al mismo tiempo, soporta los protocolos de comunicación HTTP y MQTT, ofrece una conexión WiFi y una interfaz Web para su configuración.

La interfaz web permite elegir y configurar el protocolo de comunicación de red. Dentro de esta configuración se puede configurar el periodo de emisión de las tramas, la dirección del servidor remoto y su autentificación.

Además de la configuración anterior, el gateway permite filtrar las tramas emitidas por los Beacons. Este filtro se realiza a partir de la dirección MAC del Beacon.

Middleware

También conocido como lógica de intercambio de información entre aplicaciones. Es un software que asiste a una aplicación para interactuar o comunicarse con otras aplicaciones.

Debido a sus características, no se puede establecer una comunicación directa entre el gateway y el Back-End de Neki. Se va a crear este componente como un intermediario entre el gateway y el Back-End, el middleware va a permitir conectar el gateway con el Back-End.

Este componente realiza la tarea de recibir las tramas generadas por el gateway e interpretar su información. En base a esta información se generan mensajes que corresponden a actividades dentro de la vivienda. Un ejemplo de esta actividad seria la detección de un movimiento.

Como parte de este proyecto se ha diseñado e implemetado este componente.

Back-End Neki

Aquí es donde se lleva a cabo toda la lógica del sistema. Además, este componente es el encargado de interactuar con los tutores mediante una APP Web.

A través de este componente los tutores establecen las reglas de inferencia que se deben cumplir en la vivienda del monitorizado.

El back-end utiliza la información que envía el middleware para determinar si se están cumpliendo las reglas de inferencia o no. El incumplimiento de una de estas reglas generará un aviso que será enviado al tutor.

Este componente está actualmente proporcionando el servicio de localización GPS de Neki y la comunicación con los clientes a través de APP móvil.

Cualquier modificación de este componente puede dejar todo el servicio de Neki sin funcionar provocando grandes pérdidas.

Por esta razón interesa añadir el sistema de monitorización dejando intacto el Back-End. Por otra parte, hay que modificar la APP móvil para ofrecer el servicio de monitorización.

No se va a entrar en detalles acerca de este componente porque es propiedad de la empresa Neki y se quiere mantener su confidencialidad.

3. Requisitos

En este apartado se incluyen todos los requisitos, funcionales y no funcionales, para el software que se plantea en este trabajo. Dichos requisitos son la base sobre la cual se realiza el desarrollo de la aplicación.

Cada uno de los requisitos vendrá definido por una tabla como la siguiente:

Identificador					
Descripción					
Verificabilidad	☐ Alta ☐ Media ☐ Baja	Necesidad	☐ Esencial ☐ Deseable ☐ Opcional	Prioridad	☐ Alta ☐ Media ☐ Baja

Figura 3. Plantilla tabla requisitos

A continuación, se incluye una breve descripción de los campos que componen la tabla:

- **Identificador**: Este campo corresponde al identificador del requisito. Dicho identificador será único e inconfundible, de tal forma que cada requisito sea identificable sin posibilidad de error.
  Cada uno de los identificadores seguirá la siguiente nomenclatura:
  - Requisitos funcionales: RF- y un número entre 01 y 99.
  - Requisitos no funcionales: RNF- y un número entre 01 y 99.
- **Descripción**: Campo que incluye una descripción del requisito en cuestión.
- **Necesidad**: Este campo indica la necesidad de incorporar el requisito en el sistema. Los posibles valores son:
  - **Esencial**: El requisito debe introducirse obligatoriamente dentro del sistema desarrollado.
  - **Deseable**: La incorporación de estos requisitos en el sistema puede darse en función del desarrollo del mismo.
  - **Opcional**: La incorporación del requisito en el sistema es opcional.
- **Verificabilidad:** Indica la posibilidad de comprobar que el requisito se haya incorporado al sistema. Los posibles valores son:
  - **Alta**: Se puede comprobar de forma sencilla y sin ningún tipo de dudas que el requisito está en el sistema final.
  - **Media**: La comprobación de que el requisito ha sido incorporado en el sistema no tiene una dificultad muy elevada.
  - **Baja**: Es difícil o imposible comprobar que el requisito ha sido introducido en el sistema final.
- **Prioridad**: Este campo indica el grado de prioridad con el que debe ser resuelto un requisito. Los posibles valores son:
  - **Alta**: El diseño e implementación del requisito es de carácter prioritario.
  - **Media**: El requisito se debe diseñar e implementar con prioridad media.
  - **Baja**: El requisito se debe diseñar e implementar con prioridad baja.

a. Requisitos funcionales

Identificador	RF-01					
Descripción	Para poder notificar a los tutores de que algo puede ir mal dentro de la vivienda Neki quiere recibir todas actividades de la vivienda en su plataforma.					
Verificabilidad	☐ Alta ☑ Media ☐ Baja	Necesidad	☐ Esencial ☑ Deseable ☐ Opcional	Prioridad	☑ Alta ☐ Media ☐ Baja	

Figura 4. Requisito funcional 1

Identificador	RF-02					
Descripción	Este sistema se basa en establecer unas reglas de inferencia que corresponden a actividades cotidianas. El tutor es quien mejor conocer las rutinas del monitorizado, por esta razón se quiere poder establecer las reglas de inferencia a través de la APP móvil.					
Verificabilidad	☑ Alta ☐ Media ☐ Baja	Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	Prioridad	☐ Alta ☑ Media ☐ Baja	

Figura 5. Requisito funcional 2

Identificador	RF-03					
Descripción	El tutor es quien está al cargo del monitorizado. Quiere saber en todo momento que todo va bien en la vivienda del monitorizado y ser avisado cuando algo no vaya bien para tomar las acciones oportunas.					
Verificabilidad	☑ Alta ☐ Media ☐ Baja	Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	Prioridad	☑ Alta ☐ Media ☐ Baja	

Figura 4. Requisito funcional 3

Identificador	RF-04					
Descripción	Neki y el tutor quieren conocer el porcentaje de batería de lo Beacons para conocer cuando es necesario cambiar la batería.					
Verificabilidad	☑ Alta ☐ Media ☐ Baja	Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	Prioridad	☐ Alta ☐ Media ☑ Baja	

Figura 5. Requisito funcional 4

Identificador	RF-05					
Descripción	Neki quiere detectar los Beacons que han dejado de emitir durante un tiempo configurable para poder solucionar el problema lo antes posible.					
Verificabilidad	☐ Alta ☑ Media ☐ Baja	Necesidad	☐ Esencial ☑ Deseable ☐ Opcional	Prioridad	☐ Alta ☑ Media ☐ Baja	

Figura 6. Requisito funcional 5

Identificador	RF-06				
Descripción	Una persona no autorizada puede conectarse al servicio y hacer uso de él. Para evitar esta situación Neki quiere que se procesen solo aquellas tramas de gateways que están registrados en el sistema.				
Verificabilidad	☑ Alta ☐ Media ☐ Baja	Necesidad	☐ Esencial ☑ Deseable ☐ Opcional	Prioridad	☐ Alta ☑ Media ☐ Baja

Figura 7. Requisito funcional 6

Identificador	RF-07				
Descripción	Neki va a ofrecer este producto como un servicio de alquiler. Cuando una persona se quiera dar de baja al servicio se quiere poder desactivar su gateway para no proporcionarle servicio. A su vez, cuando una persona comience a usar este servicio se quiere dar de alta su gateway.				
Verificabilidad	☑ Alta ☐ Media ☐ Baja	Necesidad	☐ Esencial ☑ Deseable ☐ Opcional	Prioridad	☐ Alta ☐ Media ☑ Baja

Figura 8. Requisito funcional 7

Identificador	RF-08				
Descripción	Para tener un feedback del sistema y un control sobre las actividades en las viviendas Neki quiere tener un registro con actividades producidas.				
Verificabilidad	☐ Alta ☑ Media ☐ Baja	Necesidad	☐ Esencial ☑ Deseable ☐ Opcional	Prioridad	☐ Alta ☐ Media ☑ Baja

Figura 9. Requisito funcional 8

b. Requisitos no funcionales

Identificador	RNF-01				
Descripción	Debido a que la plataforma de Neki está proporcionando actualmente servicio se quiere que la integración de este nuevo sistema a su plataforma se haga sin modificar la plataforma actual.				
Verificabilidad	☑ Alta ☐ Media ☐ Baja	Necesidad	☐ Esencial ☑ Deseable ☐ Opcional	Prioridad	☑ Alta ☐ Media ☐ Baja

Figura 10. Requisito no funcional 1

Identificador	RNF-02				
Descripción	Neki quiere estimar el tráfico de red móvil que se produce por el gateway y así tener una idea del coste de este servicio.				
Verificabilidad	☐ Alta ☑ Media ☐ Baja	Necesidad	☐ Esencial ☑ Deseable ☐ Opcional	Prioridad	☐ Alta ☐ Media ☑ Baja

Figura 11. Requisito no funcional 2

Identificador	RNF-03				
Descripción	Debido a su fácil instalación y larga autonomía Neki quiere usar sensores acelerómetro para detectar los movimientos de una puerta.				
Verificabilidad	☑ Alta ☐ Media ☐ Baja	Necesidad	☑ Esencial ☐ Deseable ☐ Opcional	Prioridad	☑ Alta ☐ Media ☐ Baja

Figura 12. Requisito no funcional 3

Identificador	RNF-04				
Descripción	Saber si se ha abierto una puerta mediante un sensor es mínimamente invasivo, mientras que el uso de una cámara es invasivo para el monitorizado. Este sistema quiere respetar lo máximo posible la intimidad y privacidad del monitorizado.				
Verificabilidad	☐ Alta ☑ Media ☐ Baja	Necesidad	☐ Esencial ☑ Deseable ☐ Opcional	Prioridad	☑ Alta ☐ Media ☐ Baja

Figura 13. Requisito no funcional 4

4. Decisiones de diseño

En la etapa inicial de diseño se han tomado una serie de decisiones que marcan toda la etapa de diseño e implementación, estas decisiones se han tomado a partir de los requisitos funcionales y no funcionales.

a. Message Queuing Telemetry Transport (MQTT)

El gateway proporcionado permite el usar los protocolos de comunicación HTTP y MQTT. Debido a sus características se ha optado por el uso de MQTT.

En este apartado se explica en que consiste el protocolo MQTT y que ventajas ofrece con respecto a HTTP.

MQTT es un protocolo de comunicación ligero pensado para el *Internet of things*. Está basado en la arquitectura publicador/suscriptor. Su paradigma se basa en utilizar un servidor central (broker) en donde los clientes (publicadores) envían sus mensajes a receptores específicos (subscriptores).

La arquitectura está diseñada para desacoplar la comunicación entre los nodos del sistema, esta característica permite que se puedan quitar y añadir nodos en el sistema sin la necesidad de realizar modificaciones.

En MQTT se definen dos tipos de entidades de red: un intermediario de mensajes o broker y un número de clientes.

El broker se encarga de recibir los diferentes mensajes de los clientes, estos mensajes son publicados en temas específicos o topics. Los clientes encargados de recibir las tramas se subscriben a estos temas para recibir los mensajes publicados.

Al igual que HTTP, el protocolo MQTT está basado en una comunicación TCP. Como interesa que la aplicación genere el menor tráfico de datos posible se ha decidido usar MQTT en vez HTTP.

La razón es que HTTP es un protocolo más pesado. Los paquetes de datos de MQTT contienen muy poco overhead y su payload es enviado formato binario garantizando un tráfico de red eficiente.

b. Creación de un Middleware

Uno de los requisitos no funcionales establecidos es que no se quiere modificar la plataforma actual de Neki.

Para integrar este sistema hay que establecer una comunicación entre el gateway y el Back-end de Neki.

Crear una comunicación directa entre el gateway y el back-end implica modificar uno de los dos componentes. No es posible modificar el gateway porque posee un firmware desarrollado por el fabricante que no es accesible ni modificable.

Por otra parte, modificar el back-end implica adaptarlo para que sea capaz de interpretar los datos enviados por el gateway. Esta tarea implica modificar este componente para que sea capaz de procesar las tramas del gateway, lo que supone una tarea tediosa propicia a generar errores.

Además, es importante resaltar que si se decide usar otro tipo de dispositivo como gateway va a ser necesario volver a modificar el Back-End para poder interpretar el nuevo protocolo.

A todo esto, hay que sumar que establecer una comunicación directa compromete la seguridad de la plataforma de Neki.

Como alternativa a la comunicación directa se decide crear una pasarela software o middleware. Este componente gestiona la comunicación entre el gateway y el back-end.

El middleware se va a encargar de recibir las tramas del gateway, interpretarlas y generar mensajes usando el protocolo interno de Neki. Los mensajes son enviados al Back-End de Neki y corresponden a las diferentes actividades que ocurren dentro de una vivienda.

Con respecto a la arquitectura del middleware, estará formado por dos estructuras. Por un lado, va a tener un broker encargado de recibir las tramas del gateway, por otro un cliente MQTT que va a recibir y procesar las tramas emitidas. En el otro extremo del middleware se va a llevar a cabo el protocolo de comunicación privado de Neki.

Como Neki está utilizando estas tecnologías, el middleware será desarrollado usando Node.Js y utilizará una base de datos no relacional MongoDB para la persistencia de los datos. A continuación, se muestran los motivos de su elección:

Node.Js

Node.JS es un entorno JavaScript del lado del servidor, basado en eventos. Node proporciona un entorno de ejecución que compila y ejecuta JavaScript a velocidades muy altas.

Cualquier usuario con conocimientos de JavaScript puede programar en Node.JS. Permite utilizar el mismo lenguaje tanto en el lado del cliente como en el lado del servidor.

A diferencia de JavaScript, Node.JS permite ejecutar las aplicaciones sin usar un navegador web.

Con Node.Js es posible hacer en el servidor todo lo que se necesita: acceso a fichero, base de datos, conexiones de clientes.

Se ha decidido implementar el middleware con Node.JS por los siguientes motivos:

- Rápido de programar y desplegar en un entorno de ejecución.
- Existencia de un gran número de librerías de uso libre que facilitan el desarrollo.
- Soporte para un gran número de conexiones y tráfico de manera eficiente.
- Conocimientos previos de programación en Javascript.
- El entorno de Neki está desarrollado utilizando este entorno.

Base de datos

Decidir qué tipo de base de datos usar ha sido una tarea compleja. Dentro de las opciones se ha manejado usar una de tipo SQL como MySQL o PostgreSQL, o usar una de tipo NoSQL como MongoDB.

Estudiemos las dos opciones:

MySQL es un sistema de gestión de base de datos relacional. Esto significa que guarda los datos en tablas y utiliza SQL (structured query languaje) para el acceso de esos datos. Debes predefinir el esquema de la base de datos indicando las relaciones entre los campos de las tablas.

MongoDB en cambio, guarda los datos en documentos de tipo JSON que pueden variar de estructura. La información relacionada es guardada conjuntamente para un rápido acceso mediante el lenguaje de query de MongoDB

Finalmente, la base de datos elegida fue MongoDB por los siguientes motivos:

- Se ha querido desarrollar el sistema usando esta tecnología.
- MongoDB proporciona un sistema de ficheros eficiente ante fallos y balanceo de carga.
- Tanto Node.JS como MongoDB utilizan JSON de forma nativa.

5. Diseño

Una vez establecidas las decisiones de diseño y tecnologías a utilizar se proceden a diseñar el sistema.

Debido a que muchos de los componentes ya están diseñados e implementados total o parcialmente (Beacons, gateway y back-end) el diseño e implementación van a estar centrados en el middleware.

a. Diagrama de despliegue

El siguiente diagrama muestra la estructura física del sistema, así como sus conexiones y las tecnologías de comunicación empleadas.

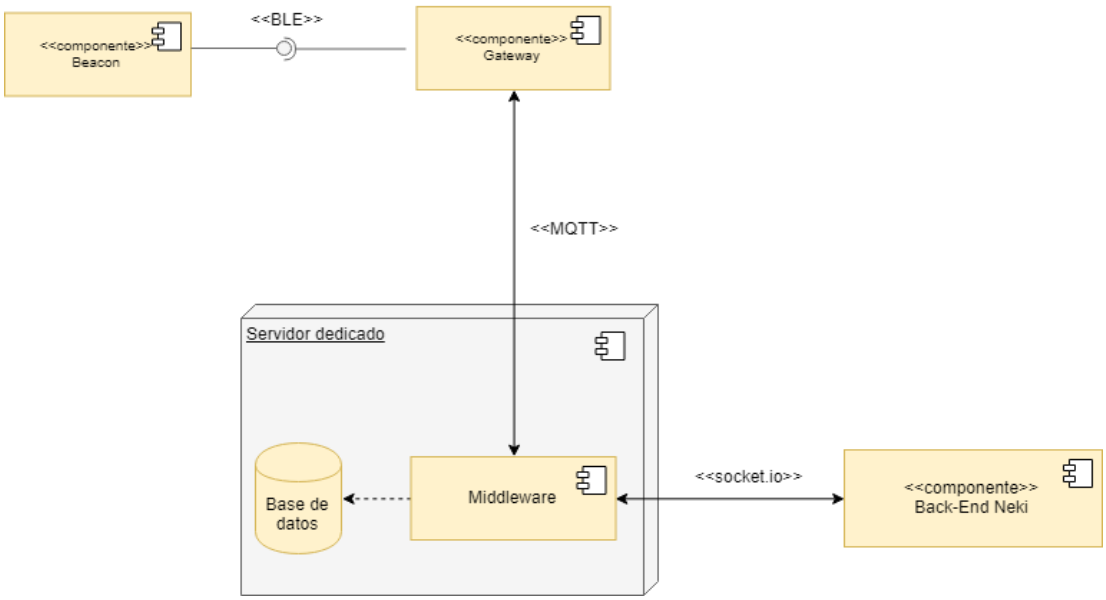

Figura 14. Diagrama de despliegue

El sistema va a estar formado por 4 componentes:

- **iBeacons**: Envía mediante BLE periódicamente la información acerca de su estado, así como la información que tiene el sensor.
- **Gateway**: Recogerá la información de todos los Beacons cercanos mediante BLE, juntará toda esa información en una trama en formato JSON y la enviará a través de la red mediante protocolo MQTT a la dirección configurada.
- **Servidor dedicado**: El servidor dedicado se compone del middleware y una base de datos, el middleware recibe las tramas del gateway, filtrando y procesando la trama en formato JSON, realiza consultas sobre la base de datos para obtener información

adicional que no transmite el gateway. Cuando los datos de un sensor generen un evento el middleware enviara este evento al Back-End.

- **Back-End**: se encarga de recibir las notificaciones del middleware y gestionar la comunicación con los tutores, notificándolos cuando algo vaya mal en la vivienda del monitorizado.

b. Diagrama de paquetes

A continuación, se muestra el diagrama de paquetes del proyecto para el middleware. En él se pueden ver las relaciones entre los componentes de la aplicación.

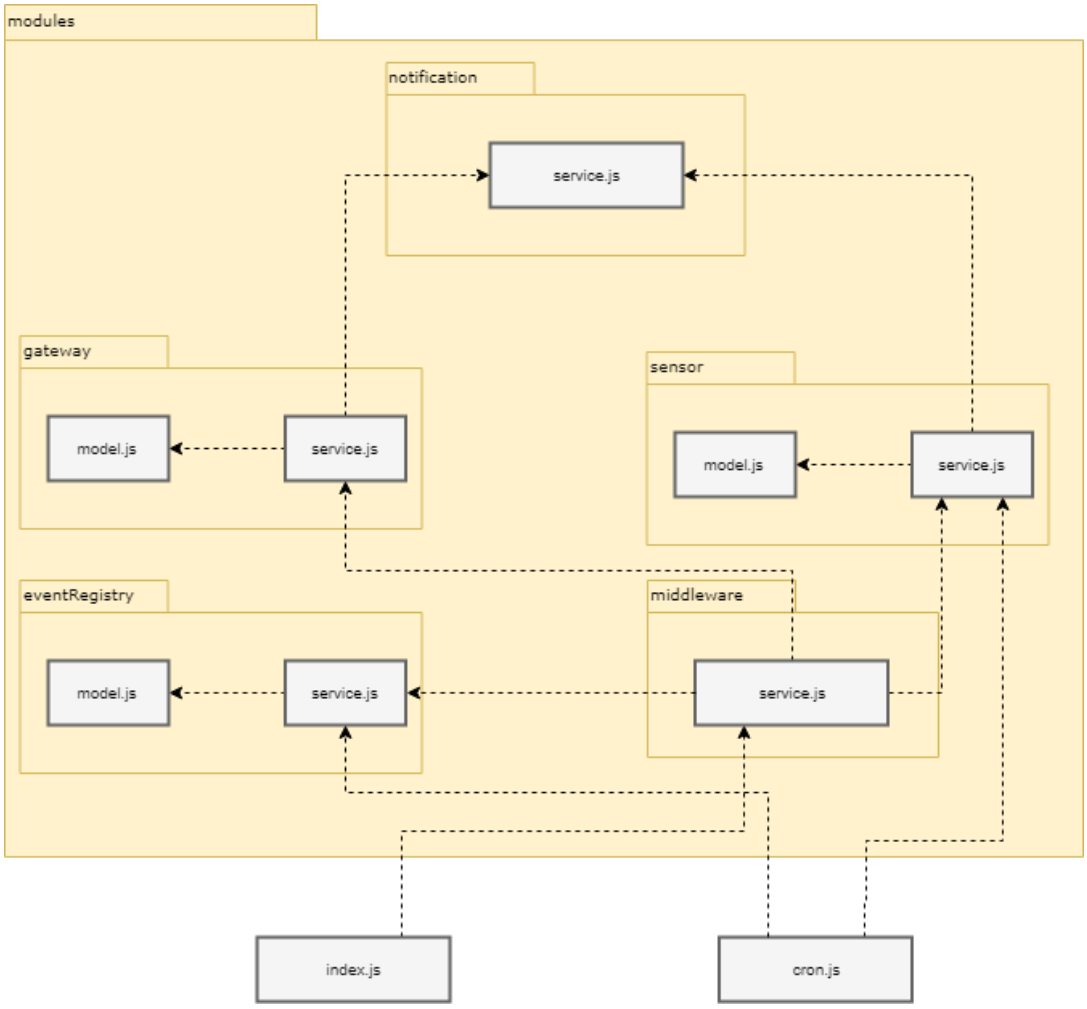

Figura 15. Middleware. Diagrama de paquetes

La organización del proyecto va a seguir un modelo en capas, este modelo está basado en el concepto de que todos los niveles de la aplicación, son una colección de componentes que se proporcionan servicios entre sí o a otros niveles adyacentes, en el caso de este proyecto va a existir una capa de datos que es donde se van a crear los modelos de acceso a los datos.

La capa de negocio va a estar por una serie de servicios mediante la cual se va a desarrollar la lógica del programa a la vez que se va a comunicar con la capa de datos para la obtención de datos.

El script index.js contendrá en punto de partida de la aplicación, en él se crea la conexión con la base de datos y se inicia el modulo que crea el servidor MQTT.

Por otro lado, el script cron.js contiene todas las tareas programadas que realice la aplicación, cada cronjob llamara a los servicios necesario para realizar las labores que necesita.

Dentro de la carpeta modules se encuentra el corazón de la aplicación, dentro de ella existirá una serie de carpetas que contienen los modelos y servicios necesarios para que la aplicación funcione, los servicios proporcionan una determinada funcionalidad, en el caso de la carpeta gateway, el fichero service.js proporciona funciones para interactuar con la tabla gateway de la base de datos mientras que el fichero model.js contiene el modelo de la base de datos para la tabla gateway.

La carpeta bróker contiene el fichero service.js que contiene el flujo principal del funcionamiento del gateway, en este script es donde se procesan los datos de las tramas enviadas por el gateway, se realizan las consultas a la base de datos y se envían las notificaciones.

En el apartado de implementación se entra en más detalles sobre la implementación del middleware.

c. Diagrama de modelo de datos

En este diagrama se muestra la estructura lógica del almacenamiento de los datos, así como sus relaciones.

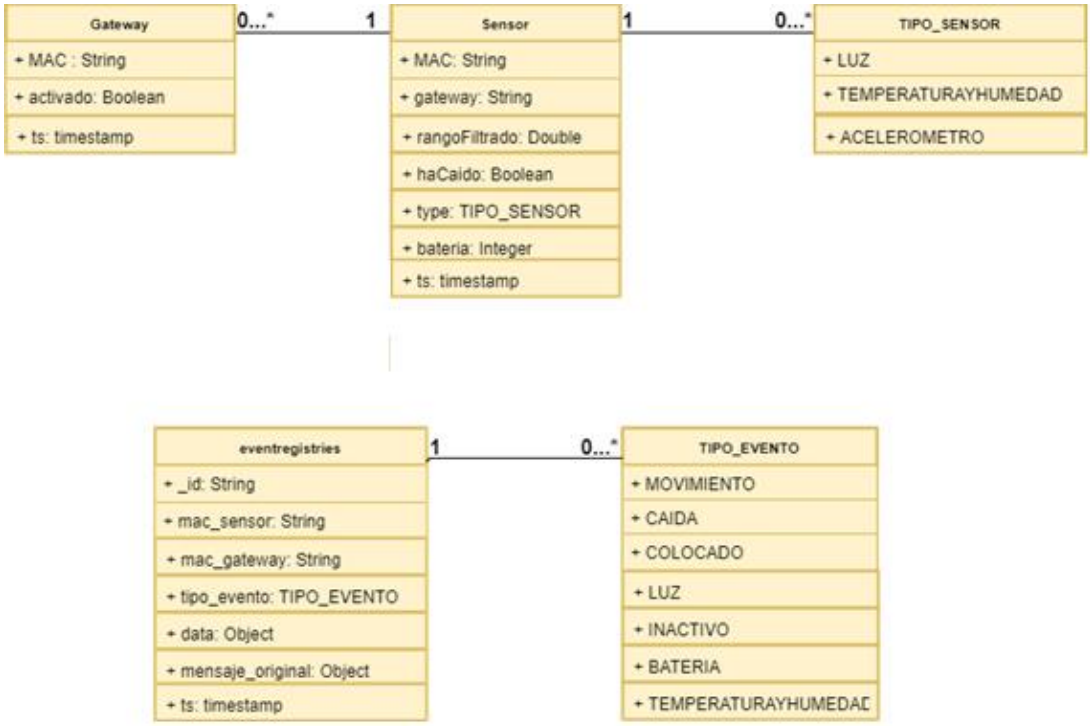

Figura 16. Middleware. Diagrama de modelo de datos

La base de datos va a estar formada por **3 tablas** y 2 tipos de datos **enumerados**:

- **Gateway**: Almacena la información de todos los gateways registrados en el middleware, cada uno de ellos se identifica mediante la MAC del dispositivo y contiene un valor booleano para saber si se pueden procesar sus tramas o no.
- **Sensor**: Almacena la información de todos los sensores registrados, en cada documento se almacena:
  La dirección MAC del dispositivo y el gateway asociado.
  Un filtro configurable de la señal emitida por el acelerómetro (el filtro permite determinar cuando los valores emitidos por el acelerómetro corresponden a un movimiento de puerta).
  Un valor booleano para conocer si este sensor ha caído o no.
  El ultimo porcentaje de batería recibido.
  Se ha dado la posibilidad de añadir más tipos de sensores: de luz y de temperatura y humedad.
- **EventRegistries**: Esta colección almacena todos los eventos producidos en el sistema, es un log del sistema para poder depurar posibles fallos. En esta tabla se almacenará un _id generado por mongo, la dirección MAC del sensor que ha producido el evento, así como que datos han producido ese evento (en el caso del acelerómetro el valor de aceleración del eje z), por último, se almacenara la trama que ha producido el evento.

6. Implementación del middleware

A continuación, se muestra cómo se ha implementado el middleware. Este apartado explica la distribución del código y los conceptos fundamentales. No se explica ninguna línea de código ya que este no es el objetivo de este trabajo, sin embargo, si se van comentar de los conceptos importantes, por ejemplo, como se inicializa este componente.

Para desarrollar el middleware se ha utilizado el IDE Webstorm 2017 v1.2, node.JS v4.6, el gestor de módulos npm v2.15.9 y MongoDB v3.6.

El código está alojado en un repositorio privado, debido a que va a ser un producto comercial no interesa que personas ajenas al proyecto accedan al código. Neki es quien tiene permiso para acceder y modificar este código.

El middleware ha sido diseñado e implementado siguiendo el patrón de diseño modelo-vista-controlador(MVC)[2]. Este patrón separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.

Al no poseer una interfaz de usuario la capa de vista no se ha implementado.

A continuación, se entra en detalle sobre como ha quedado implementado el middleware:

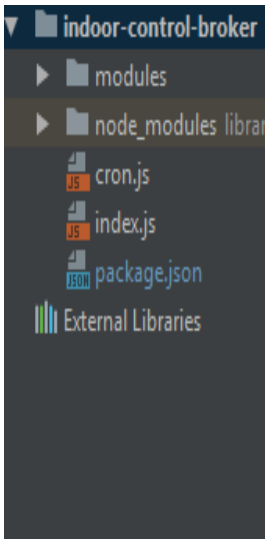

Figura 17. Directorio raíz de la aplicación de node.js

Como cualquier aplicación de Node.js, en el archivo index.js están las declaraciones para que pueda iniciar el middleware: inicialización del servidor MQTT, declaración de variables de entorno, rutas, conexión a la base de datos…

Ejecutando el comando ***node index.js*** el middleware se pone en ejecución.

Otros archivos:

Package.json: contiene todas las dependencias de la aplicación, se debe instalar todas antes de iniciar la aplicación utilizando el comando **npm install**.

Cron.js: contiene todas las tareas que se van a realizar automáticamente de manera periódica.

En el directorio node_modules se guardan todas las librearías dependientes del proyecto.

[2] https://si.ua.es/es/documentacion/asp-net-mvc-3/1-dia/modelo-vista-controlador-mvc.html

Figura 18. Subdirectorio '/modules'

El directorio module es el que contiene los modelos y controladores del patrón MVC junto con la lógica de la aplicación.

Se han creado los siguientes modelos de datos:

Gateway: Contiene los atributos del gateway como por ejemplo la dirección MAC.

Sensor: Contiene los atributos asociados a un sensor.

EventRegistry: Contiene los atributos necesarios para almacenar todas las actividades que se producen en el middleware.

Para cada uno de estos modelos de datos se ha creado un servicio que interactuar con la capa de modelo, insertando, modificando o borrando registros de datos.

El directorio **middleware** contiene toda la lógica del middleware, en él se procesa la información de los sensores y se llaman a las funciones de notificación.

El directorio **notification** contiene las funciones necesarias para gestionar la comunicación con el back-end de Neki usando su protocolo.

7. Evaluación

A continuación, se evalúan los beacons y el gateway. Esta evaluación pretende conocer la validez de estos Beacons en nuestro sistema.

Primero se realiza un estudio que determine si los beacons proporcionados por Neki son capaces de detectar los movimientos de unas puertas. Además, se va a evaluar si el uso de un disparador no configurable es capaz de detectar estos movimientos.

Posteriormente se va a evaluar y estimar el tráfico de datos producido por el gateway usando el protocolo MQTT. Por ser un protocolo más pesado, HTTP queda descartado de esta evaluación.

a. Beacons con acelerómetro para la detección de movimiento en puertas

Tal y como se ha explicado en la sección de arquitectura, los beacons emiten sus tramas con una determinada frecuencia, esta frecuencia es configurable a través de una APP móvil.
Además, se puede activar un disparador o trigger de manera que emitan su información cuando se produce un evento, en el caso de estos Beacons el evento es detectar una aceleración.

El modo trigger solo emite tramas cuando el acelerómetro detecta una aceleración. Permitiendo un mayor ahorro de batería, y por lo tanto duración que no usándolo.

El fabricante no permite configurar la sensibilidad de este disparador. La sensibilidad preconfigurada por el fabricante puede no ser lo suficientemente sensible como para detectar el movimiento de una puerta.

No usar este disparador hace que el Beacon tenga un mayor tráfico y consumo de batería, configurando la frecuencia de muestreo se puede llegar a un compromiso precisión-consumo de batería.

Una frecuencia baja va a generar más datos y va a ser más fácil detectar el movimiento de una puerta, pese a un mayor consumo de batería. Por el contrario, una frecuencia alta va a ahorrar más batería, pero es posible que no le dé tiempo a detectar el movimiento.

En este apartado se va a realizar un estudio del funcionamiento de los Beacons. Este estudio busca conocer si el uso de un disparador es capaz de detectar el movimiento de una puerta.

El estudio va a analizar tres tipos de puertas: La puerta de una nevera, la puerta de una habitación y la puerta de casa.

Entorno de pruebas

A continuación, se describe el entorno de pruebas desarrollado para realizar el estudio. Este entorno busca extraer los valores que están generando los Beacons en el instante en el cual se está abriendo y cerrando una puerta.

Para poder determinar el instante en el cual se esa abriendo o cerrando una puerta se va a usar un sensor mecánico.

El entorno de pruebas va a estar formado por los siguientes componentes:

- **Beacons**: Son las balizas que contienen el sensor acelerómetro. Se van a usar dos Beacons, uno usando el disparador y otro sin él.
- **Sensor mecánico**: Final de carrera usado para determinar si una puerta está abierta o cerrada de manera fiable.
- **Gateway**: Se encarga de enviar los datos al recolector de datos mediante MQTT.
- **Recolector de datos**: Consiste en una pequeña aplicación que se va a encargar de recoger los datos de los Beacons y del sensor mecánico a través de MQTT, para posteriormente escribir los datos generados en un fichero. Para cada Beacon se va a generar un fichero.
- **Ficheros de datos**: Este fichero contiene los valores generados por un sensor acelerómetro, el valor del sensor mecánico en un preciso instante de tiempo y una marca de tiempo.

Se ha diseñado un prototipo para poder conocer de una manera fiable cuando una puerta está abierta o cerrada. Este prototipo va a actuar como un Beacon a diferencia que va a emitir su información directamente a través de MQTT y no BLE.

La infraestructura de este prototipo va a estar formada por un final de carrera, una placa NodeMCU ESP2866 y una batería.

Con el uso de un final de carrera, se puede estar seguro de cuando una puerta está abierta o cerrada y, conociendo el instante en el cual se produce este evento, se puede conocer la señal generada por el acelerómetro.

Para poder comunicar este final de carrera con el recolector de datos se ha usado una placa NodeMCU ESP2866[3]. Esta placa de código abierto ofrece soluciones para aplicaciones que requieren una conectividad Wi-Fi.

[3] https://electronilab.co/tienda/nodemcu-board-de-desarrollo-con-esp8266-wifi-y-lua/

En la siguiente imagen se muestra el prototipo creado para llevar a cabo las pruebas:

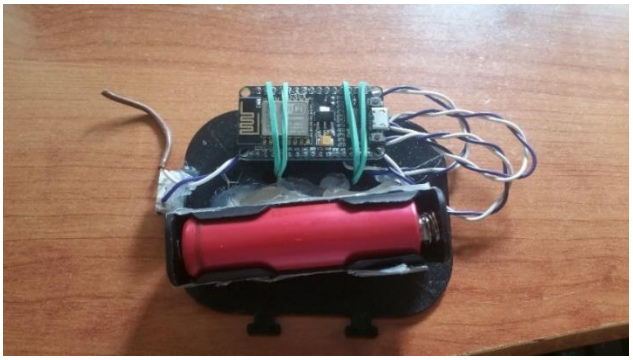

Figura 19. Sensor mecánico

Se ha programado esta placa para que sea capaz de realizar publicaciones sobre el recolector de datos a través de MQTT. Enviando cada segundo el valor del final de carrera (0 si la puerta está cerrada y 1 si está abierta).

Esta placa va a estar alimentada por una batería para evitar el uso de cableado y así poder colocar este dispositivo en casi cualquier lugar.

La siguiente imagen muestra la estructura final del entorno de pruebas:

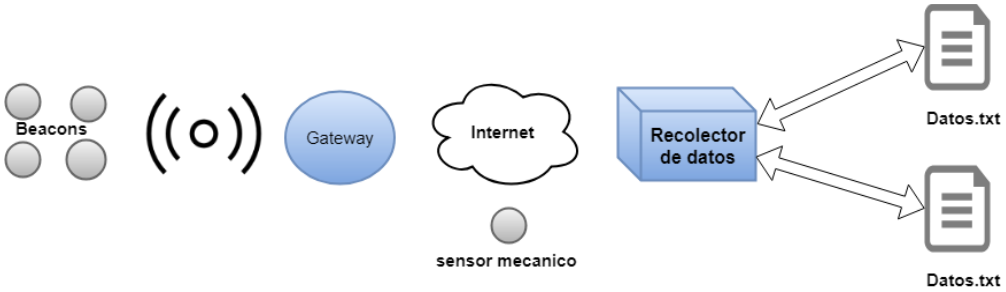

Figura 20. Entorno de pruebas

Para visualizar los datos de los ficheros generados se van a usar gráficas.
Las gráficas van a mostrar los datos generados por del acelerómetro cuando la puerta está en uno de los siguientes estados: apertura, reposo y cierre.

En los siguientes apartados se muestran los resultados obtenidos para cada una de las puertas mencionadas anteriormente.

ANALISIS DEL MOVIMIENTO DE PUERTA NEVERA

Una manera de detectar que algo no va bien en la vivienda es conociendo que durante un determinado tiempo el monitorizado no ha hecho uso de la nevera.

Aunque sería lo idóneo, en este caso no es tan relevante saber si puerta de la nevera está abierta o cerrada desde el punto de vista del monitorizado. En este proyecto es más

relevante el hecho de que se haya movido la nevera que se haya quedado abierta y se queme el motor.

En este experimento se busca obtener el menor número de falsos positivos posibles. Por falso positivo se entiende que se haya detectado un movimiento cuando en realidad no se ha producido. Este falso negativo indicaría que se ha producido una actividad y por lo tanto todo va bien en la vivienda cuando no es así.

En esta sección se muestran los resultados obtenidos al realizar cuatro aperturas y cierres de la puerta de una nevera.

El experimento ha consistido en la colocación de dos Beacons (uno con disparador y otro sin él) y un final de carrera colocado en la puerta de una nevera.

Este sensor mecánico detectara el instante en el cual la puerta de la nevera se abre y se cierra

Para poder interpretar mejor los datos generados se han creado una serie de gráficas.

Dentro de estas gráficas, el eje x representa el tiempo, el eje y muestra los datos generados por los sensores a lo largo del tiempo. Para cada una de las gráficas se van a mostrar dos señales:

La señal azul representa el valor producido por el acelerómetro en el eje Z. Este valor corresponde a una magnitud de fuerza G.

Únicamente se ha analizado el eje Z porque es el único eje que es sensible al movimiento de la puerta.

En la imagen de la derecha se muestra como quedan las coordenadas de los Beacons cuando se coloca en una puerta.

Las puertas se abren y se cierran horizontalmente, según la orientación del Beacon este movimiento solo va a implicar al eje Z.

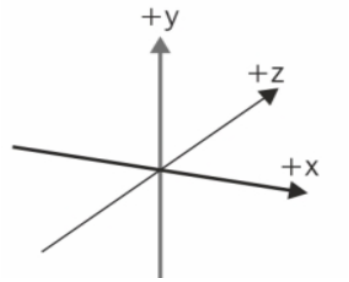

Figura 21. Orientación Beacon

La señal naranja representa el valor generado por el sensor mecánico a lo largo del tiempo. El valor 0 representa que la puerta está cerrada en ese momento, el valor 0.15 implica que la puerta está abierta. Se ha usado el valor 0.15 y no 1 para poder visualizar mejor las gráficas.

Para mostrar los resultados de este experimento se han creado unos gráficos a partir de los datos generados.

La primera grafica muestra el comportamiento del Beacon en modo muestreo, en la segunda se muestra el comportamiento del Beacon con el disparador activado.

En la siguiente grafica muestran los resultados de obtenidos sin el uso de disparador:

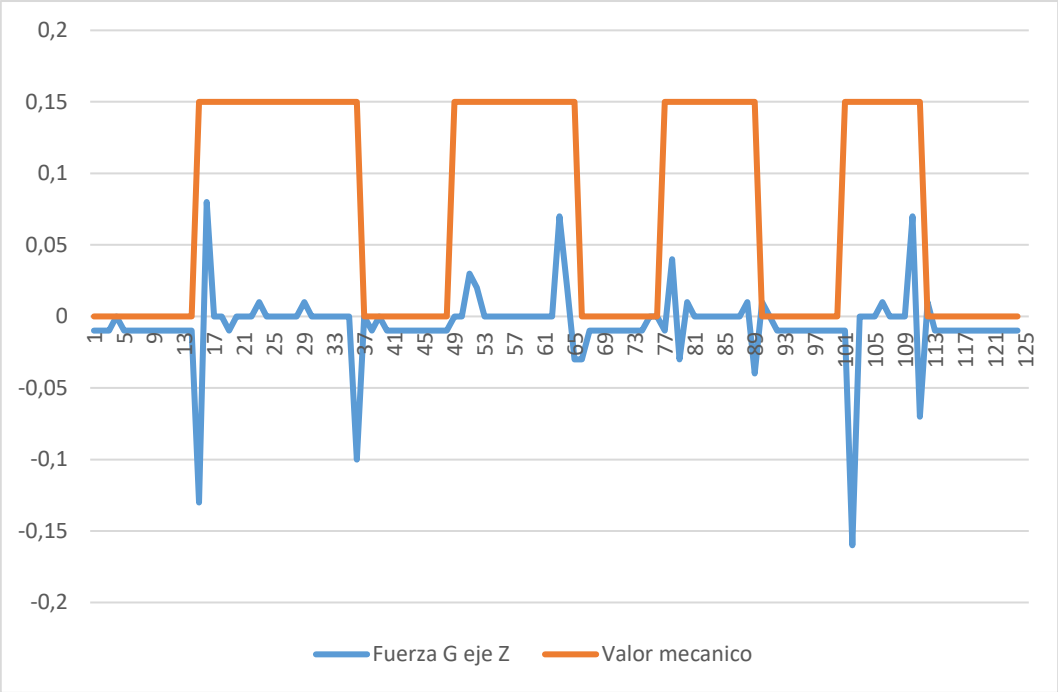

Figura 22. Grafica estudio puerta nevera sin disparador

Si nos fijamos en la señal del sensor mecánico, se observa que en los instantes 13,49,77 y 101 se han producido aperturas en la puerta. Por el contrario, en los instantes 37,65,89 y 113 se han producido cierres de la puerta.

Durante estos instantes de tiempo se ve como la señal del acelerómetro genera una serie de picos. Estos picos representan la aceleración producida en el momento de abrir y cerrar una puerta.

Los picos son variables debido a que la fuerza aplicada en cada el movimiento no es la misma. Además, los valores positivos y negativos representan la dirección de la aceleración en ese momento.

Durante los instantes 13,65,77 y 113 se ve cómo los picos cambian de signo de forma repentina. En estos casos, el cambio de signo representa el golpe producido al cerrar y abrir la puerta.

Se puede observar como no hay falsos positivos entre las aperturas de las puertas.

Por último, se puede ver que durante las aperturas 1 y 4 se produce un pequeño ruido en la señal. Esto se debe a las vibraciones generadas por el motor de la nevera. Filtrando este ruido se puede conseguir detectar el movimiento de la nevera y no obtener ningún falso positivo.

En la siguiente grafica se muestran los resultados obtenidos con el disparador activado:

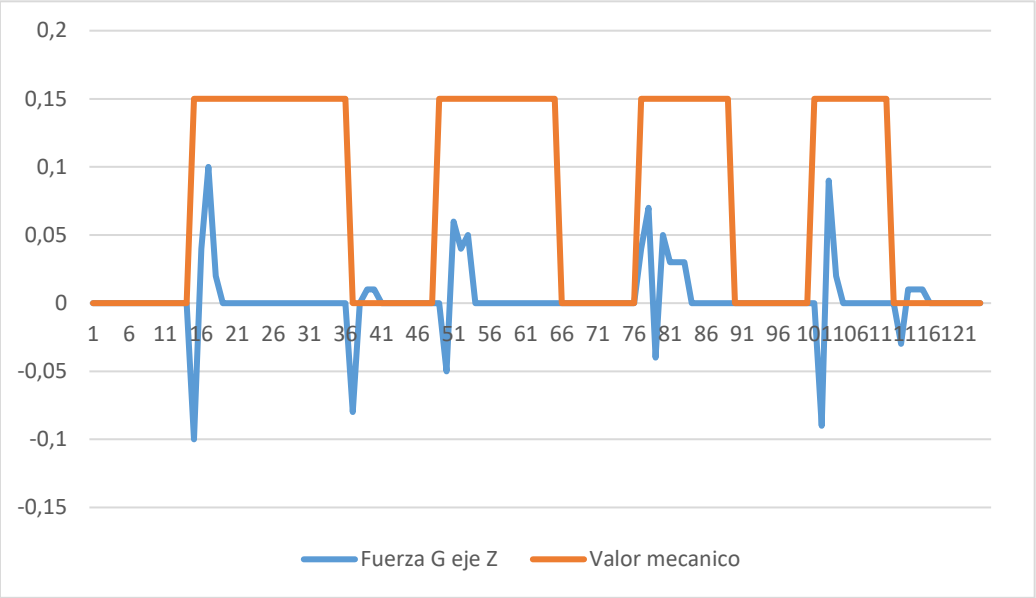

Figura 23. Grafica estudio puerta nevera con disparador

Se puede observar que cuando la puerta está en reposo no se está emitiendo ningún tipo de señal por parte del acelerómetro. Únicamente se está emitiendo la señal cuando se produce la apertura o cierre de la puerta.

En este caso, la señal del acelerómetro ha generado picos durante los instantes 13,37,49,77,101 y 113. Se observa como estos picos de señal son variables y tienen cambios de signo de manera repentina, al igual que en la gráfica anterior.

Durante los instantes 65 y 89 no se han producido picos en la señal del acelerómetro. No se ha superado el umbral que activa el evento y, por lo tanto, no se ha detectado el movimiento. El mayor valor amplitud registrada ha sido de 0.1G.

Con el simple hecho de recibir una trama se podría detectar el movimiento de una puerta, pero no se han conseguido detectar algunos movimientos, usar este disparador es poco fiable.

Este modo de funcionamiento ha detectado 6 de los 8 movimientos producidos.

ANALISIS DEL MOVIMIENTO DE LA PUERTA DE UNA HABITACION

Otra manera de detectar que algo no va bien en la vivienda es sabiendo que el monitorizado no ha entrado o salido de su habitación durante un determinado tiempo.

Al igual que en caso de la nevera, interesa saber si se ha producido un movimiento, no el estado de la puerta.

También se busca el menor número de falsos positivos posibles. Además, tiene que ser capaz de detectar los movimientos parciales de la puerta. Es interesante detectar los pequeños movimientos de la puerta porque esto quiere decir que hay actividad dentro de casa.

En esta sección se muestran los resultados obtenidos al realizar cuatro aperturas y cierres de la puerta de una habitación.

Al igual que la nevera, el experimento ha consistido en la colocación de dos Beacons en la puerta (uno muestreando cada segundo y otro muestreando por umbral) y la colocación de un final de carrera.

Para mostrar los resultados de este experimento se han creado unos gráficos a partir de los datos generados.

Estas graficas representan la misma información que en el experimento de la nevera, en el eje x se representa el tiempo, en el eje y se muestran los datos generados por los sensores.

En la siguiente grafica muestran los resultados obtenidos para el modo de emisión sin el uso del disparador:

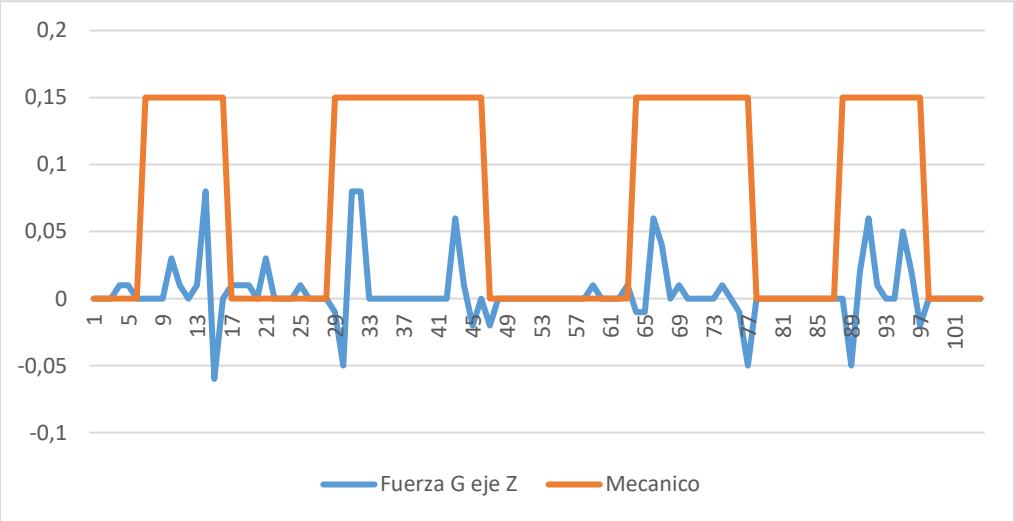

Figura 24. Grafica estudio puerta habitación sin disparador

Si nos fijamos en la señal del sensor mecánico, se observa que en los instantes 5,29,65,89 se han producido aperturas. Del mismo modo, en los instantes 17,45,77,97 se han producido sus correspondientes cierres.

Durante estos instantes se ve como la señal del acelerómetro genera una serie de picos. Estos picos representan la aceleración producida en el momento de abrir y cerrar la puerta de la habitación.

La amplitud de los picos producidos es menor que en el caso de la nevera. La causa es que en la nevera hay una goma que hace resistencia a la hora de abrir y cerrar la puerta, siendo necesario realizar más fuerza para abrir la puerta. El mayor valor de pico registrado ha sido 0.08 G.

En la primera apertura y cierre el acelerómetro ha detectado el movimiento después de que el sensor mecánico detectara la apertura de la puerta. Esto no quiere decir que no

se haya detectado el movimiento, simplemente se ha detectado la aceleración durante el proceso de apertura.

Siendo este movimiento más sensible que en el caso de la nevera, filtrando este ruido se puede conseguir detectar el movimiento de la puerta habitación.

Por último, se puede observar cómo no se han producido falsos positivos entre las aperturas y cierres de la puerta.

En la siguiente grafica se muestran los resultados obtenidos usando el disparador:

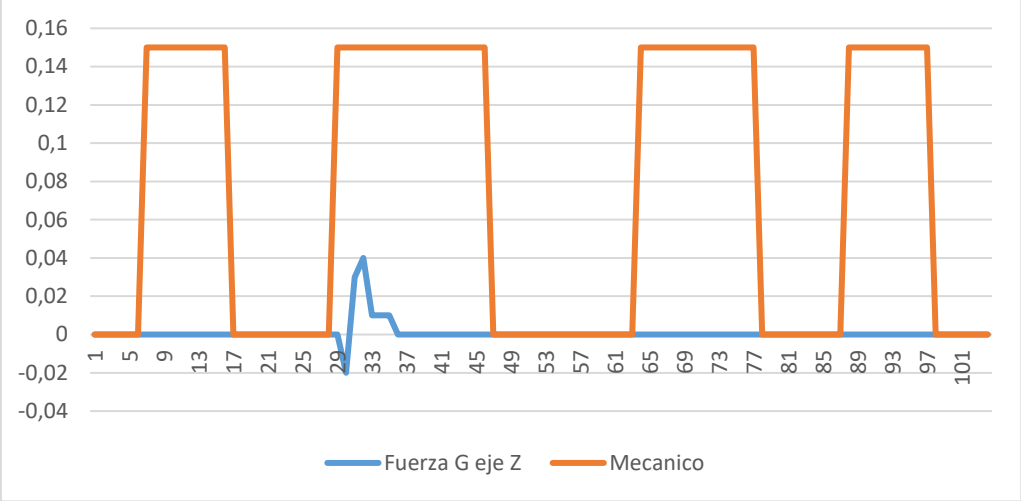

Figura 25. Grafica estudio puerta habitación con disparador

En este caso, la señal del acelerómetro ha generado picos únicamente en el instante 29. Para el resto de movimientos no se ha producido ningún tipo de señal.

Como se ha dicho en el análisis sin disparador, la aceleración generada por el movimiento de la puerta de la habitación es menor que en el caso de la nevera. El hecho de que la aceleración necesaria para mover la puerta sea menor ha provocado que se detecten menos movimientos.

En este experimento solo se ha activado el disparador una vez. Ha detectado 1 de los 8 movimientos producidos.

ANALISIS DEL MOVIMIENTO DE LA PUERTA DE CASA

En algunas ocasiones el monitorizado va a ser una persona que puede desorientarse en la calle y necesitar ayuda. En este caso, esta persona no debería salir de casa sin la supervisión de alguien.

Colocando uno de estos sensores en la puerta de casa podemos saber si el monitorizado ha salido de casa, pudiendo ponernos en contacto con él para comprobar que está bien.

A diferencia del resto de experimentos, en este caso sí que sería ideal saber si una puerta está abierta o cerrada. Se debería poder avisar al tutor de que la puerta de casa lleva un cierto tiempo abierta.

En este caso no se debería tolerar ningún falso positivo. Se debe estar seguro de que cuando se ha detectado un movimiento este se ha producido de verdad.

Dentro de esta sección se muestran los resultados obtenidos al realizar tres aperturas y cierres de la puerta de casa.

Al igual que en el resto de experimentos, se han colocación dos Beacons (uno muestreando cada segundo y otro muestreando por umbral) y un final de carrera en la puerta de casa.

A diferencia del resto de experimentos, en este interesa que no existan falsos positivos. No interesa que el monitorizado salga de casa si no debe salir o, si puede, que se deje la puerta de casa abierta.

Para mostrar los resultados de este experimento se han creado unos gráficos a partir de los datos generados.

Estas graficas representan la misma información que en otros experimentos, en el eje x se representa el tiempo y en el eje y los datos generados por los sensores.

En la siguiente grafica muestran los resultados obtenidos sin usar el disparador:

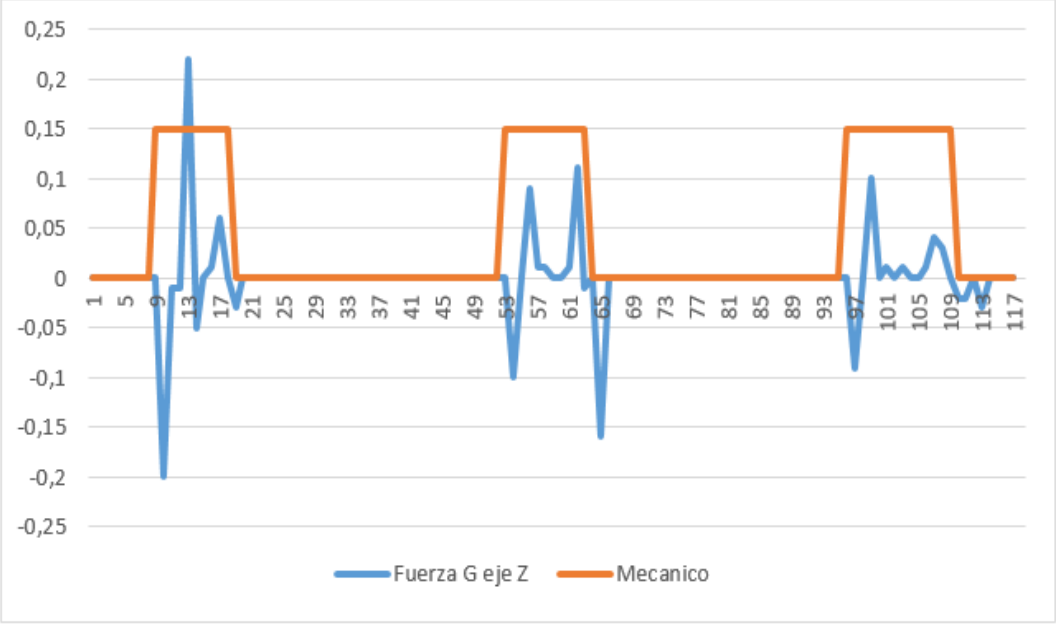

Figura 26. Grafica estudio puerta casa sin disparador

Análogamente a otros experimentos, se observa que en los instantes 9,53 y 97 se han producido aperturas de la puerta. Por otro lado, en los instantes 17,65 y 109 se han producido sus correspondientes cierres.

Durante estos instantes de tiempo se ve como la señal del acelerómetro genera una serie de picos. Estos picos representan la aceleración producida en el momento de abrir y cerrar una puerta.

Los resultados obtenidos se asemejan más al movimiento de la puerta de la nevera que al movimiento de la puerta de la habitación. La mayoría de picos alcanzan una aceleración superior a 0.1 G.

Por último, se puede ver que durante las aperturas 3 se produce un pequeño ruido en la señal. Filtrando este ruido se puede conseguir detectar el movimiento de la puerta de casa sin obtener falsos positivos.

En la siguiente grafica se muestran los resultados obtenidos usando el disparador:

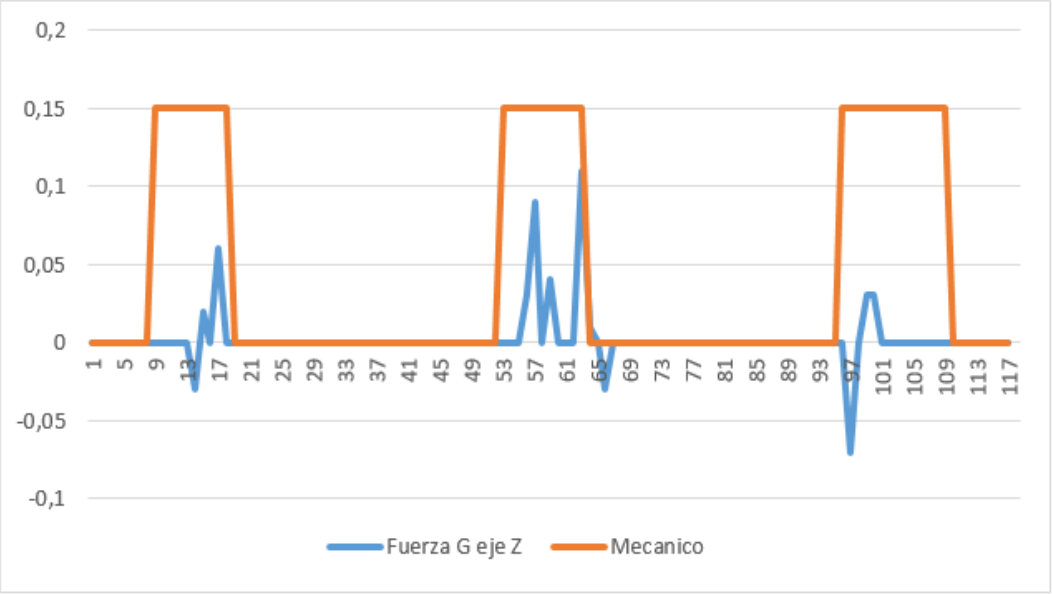

Figura 27. Grafica estudio puerta casa con disparador

En este caso, la señal del acelerómetro ha generado picos en los instantes 17, 57, 61 y 97. Para el resto de movimientos no se ha producido ningún tipo de señal.

En este caso se han detectado un mayor número de movimientos que en el caso de la puerta de la habitación.

No se ha conseguido detectar el movimiento de la primera apertura y el tercer cierre. Si el monitorizado se “escapa” de casa este modo no asegura la detección de esta actividad. A su vez, si el monitorizado se deja la puerta abierta de casa tampoco se asegura su detección.

Este modo de funcionamiento ha detectado 4 de los 8 movimientos producidos.

CONCLUSIONES

Con las gráficas obtenidas en cada uno de los experimentos se puede concluir que el uso del disparador no es lo suficientemente fiable, sin embargo, emitiendo la señal cada segundo si se pueden detectar los movimientos de la puerta.

En todos los experimentos el uso del disparador no ha llegado a ofrecer una fiabilidad del 100%. Si fuese posible configurar el umbral de activación es muy probable que su uso fuera valido.

A pesar de no estar documentado por el fabricante, según las gráficas obtenidas con disparador se puede intuir que produce un evento cuando la diferencia de aceleración es mayor o igual que 0.1G.

Sin el uso de este disparador se ha conseguido detectado todos los movimientos producidos. Por el contrario, existe un cierto ruido en la señal que debe ser filtrado para poder detectar estos movimientos. Este ruido es distinto en cada una de las puertas.

A pesar de que no usar el disparador consume más energía no supone un gran consumo. Durante un largo periodo de uso, más de tres meses, no se ha producido un descenso en el porcentaje de batería.

Aunque sería interesante, debido a esta larga duración no se ha probado a usar otras frecuencias mayores en esta evaluación.

Para poder detectar las actividades correspondientes a movimientos de puertas se va a utilizar un filtrado de la señal, este filtrado va a ser configurable. Su valor por defecto va a ser 0.08 G.

Por último, el uso del acelerómetro va a servir para detectar los movimientos de la puerta. No va a servir para saber si la puerta está abierta o cerrada.

En el caso de la puerta de casa, interesa saber si está abierta o cerrada la puerta, no se recomienda el uso estos sensores acelerómetros. Se recomienda el uso de un tipo de sensor más fiable, por ejemplo, mecánico o magnético.

b. Análisis y estimación del tráfico de datos producido por los Beacons

El gateway requiere del uso de conexión a internet para comunicarse con el middleware. Para proporcionarle esta conexión se va a utilizar una red móvil.

Actualmente Neki tiene contratado un servicio de red móvil para sus dispositivos. El proveedor de este servicio cobra una tarifa mensual por consumo de datos.

Neki quiere tener una estimación del tráfico de datos mensual para así tener una idea del gasto que va a suponer este servicio.

En este apartado se va a analizar el trafico producido entre un gateway y el middleware usando el protocolo MQTT.

Debido a que las tramas emitidas por gateway van a depender del número de sensores se van a realizar dos experimentos: En el primero se va a estimar el consumo usando un solo Beacon mientras que en el segundo se va a estimar usando cuatro Beacons.

Una vez realizados los experimentos se va a determinar la cantidad de datos generados por un sensor de media durante un segundo, minuto, hora, día y mes.

ENTORNO DE PRUEBAS

A continuación, se describe el entorno de pruebas desarrollado para realizar el estudio.

Para realizar las pruebas se va a modificar el middleware para que sea capaz de obtener el número de bytes de la trama recibida. Estas tramas serán emitidas por el gateway cada segundo. El tamaño tramas se va a almacenar en un fichero para su posterior análisis.

Este entorno buscar analizar el tamaño de las tramas emitidas por el gateway durante un minuto. Para calcular el consumo, se va a calcular el número de bytes de las tramas emitidas.

Se va a configurar el gateway para que emita tramas cada segundo, no se ha probado a calcular el consumo con otros periodos de emisión más altos porque el sistema no ha sido capaz de detectar las actividades de movimiento de las puertas.

Una vez obtenido el tamaño de las tramas emitidas durante un minuto se va a calcular su media. Esta media dará una estimación del trafico producido por minuto. A partir de esta media se va a calcular el consumo por segundos, minutos, horas, días y meses.

En los posteriores apartados se muestran los resultados obtenidos para cada una de los experimentos.

CONSUMO PRODUCIDO POR UN BEACON

Se ha ejecutado el entorno de pruebas descrito anteriormente asociando al gateway únicamente un Beacon. Los resultados obtenidos han sido representados en una gráfica.

En la gráfica, el eje x representa el tiempo mientras que el eje y representa el consumo de datos que se ha producido en ese instante.

Las gráficas muestran dos tipos de señales: la señal constante naranja representa la media del consumo producido. La señal azul representa el tamaño de la trama que ha recibido el middleware en ese instante de tiempo.

La siguiente grafica muestra el tamaño de las tramas recibidas por el middleware durante un minuto cuando el gateway tiene asociado un Beacon.

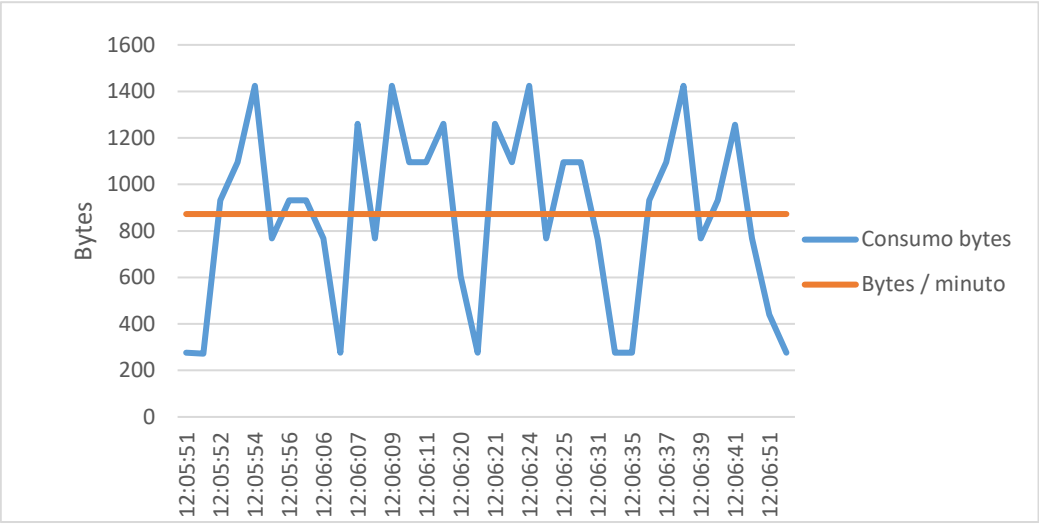

Figura 28. Grafica tráfico de datos para 1 sensor

En la gráfica se puede observar como el flujo de datos varía entre 300 Bytes y 1400 Bytes.

Esta diferencia es debida a que las tramas emitidas por el gateway contienen diferentes tipos de datos adicionales: como puede ser un timestamp, la dirección MAC del gateway, etc. Para estimar el coste no es necesario entrar en detalles sobre estas tramas.

El trafico medio por minuto obtenido es de 898 Bytes, con esta primera información podemos conocer el tráfico que se va a producir durante diferentes magnitudes de tiempo.

En la siguiente tabla se muestra la estimación del consumo de datos del sistema en diferentes magnitudes de tiempo:

	Segundo	Minuto	Hora	Día	Mes
Consumo	15 B	898 B	54 KB	1,3 MB	40,3 MB

Figura 29. Tabla tráfico de datos estimado para 1 sensor

CONSUMO PRODUCIDO POR CUATRO BEACONS

Este experimento sigue el mismo procedimiento que el anterior. Con la diferencia de que se han asociado al gateway cuatro beacons.

La siguiente grafica muestra el tamaño de las tramas recibidas por el middleware durante un minuto cuando el gateway tiene asociado cuatro Beacon.

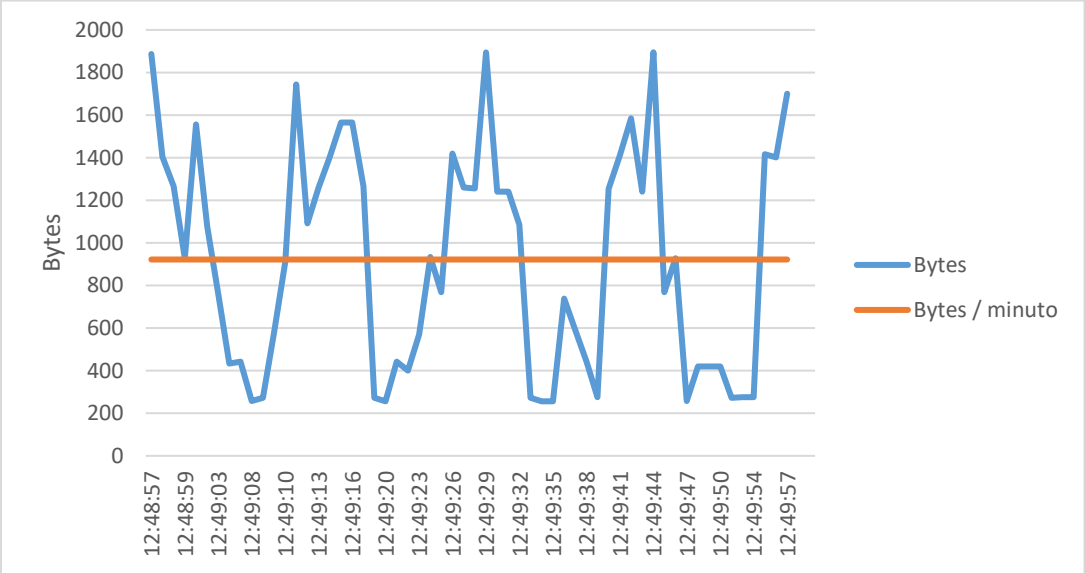

Figura 30. Grafica tráfico de datos para 4 sensores

Al igual que en la gráfica anterior, se puede observar como el flujo de datos varía a lo largo del tiempo. Con la diferencia de que los tamaños de trama varían entre 300 Bytes y 1900 Bytes. Hay una diferencia máxima de 500 Bytes con respecto al uso de un Beacon.

En este caso el tráfico medio de datos por minuto es de 922 Bytes. Se produce una diferencia de datos de apenas 30 Bytes con respecto al uso de un Beacon. Cuando se agrega un nuevo sensor el gateway únicamente envía los campos del sensor sin generar ningún tipo de overhead adicional, esta característica hace que se genere muy poco trafico.

En la siguiente tabla se muestra la estimación del consumo de datos del sistema en diferentes magnitudes de tiempo para 4 sensores:

	Segundo	Minuto	Hora	Día	Mes
Consumo	16 B	922 B	56 KB	3,4 MB	105,4 MB

Figura 31. Tabla tráfico de datos estimado para 4 sensor

CONCLUSIONES

Con los resultados obtenidos se puede decir las tramas emitidas por el gateway mediante protocolo MQTT no genera un tráfico de datos excesivo. Añadir 3 sensores al sistema ha supuesto un consumo adicional de 60 MB al mes.

No se ha proporcionado información acerca de los precios del proveedor de red de Neki. Pero según los datos obtenidos este servicio no va a suponer un gasto importante para Neki.

Se puede concluir que va a existir un consumo mínimo de 40 MB al mes y el coste de añadir un Beacon va a suponer un tráfico adicional de 21 MB al mes de media.

8. Pruebas y validación

Implementado el middleware, se ha realizado una primera fase de pruebas probando el correcto funcionamiento del middleware. Posteriormente, se ha proporcionado el middleware a Neki para que lo integre en su plataforma.

Durante esta integración se ha realizado una segunda fase de pruebas sobre el sistema final.

Este apartado explica cómo se han llevado a cabo estas pruebas. A continuación, se muestran las pruebas realizadas para validar los requisitos establecidos, tanto los funcionales como los no funcionales.

a. Pruebas iniciales

Antes de realizar las pruebas finales se ha probado que el middleware es capaz de detectar las actividades de una vivienda.

Las pruebas se han desarrollado dentro en la vivienda del autor del TFG.

Dentro de la vivienda se han colocado una serie de Beacons y un gateway. Estos Beacons se han colocado en la puerta de la nevera, la puerta del baño y la puerta de la habitación. Posteriormente se ha ejecutado el middleware en un ordenador personal y se ha configurado el gateway para que envié las tramas al middleware.

Para comprobar que el middleware es capaz de detectar y emitir las actividades de la vivienda se ha mostrado por pantalla las actividades generadas. Estas actividades van a consistir en detectar movimientos de puerta, informar de la batería…

Las pruebas realizadas han buscado:

- Probar que el middleware es capaz de recibir las tramas del gateway.
- Saber que los movimientos de las puertas se detectan correctamente.
- Obtener el porcentaje de batería de los Beacons.
- Conocer cuando un sensor está realmente inactivo.

b. Pruebas finales

Las pruebas finales se van a realizar en el entorno de desarrollo de Neki.

El middleware se va a instalar en un servidor de desarrollo de Neki. Dentro de este servidor se van a realizar las pruebas finales del sistema. Cuanto se hayan validado estas pruebas se tendrá una primera versión de producción accesible para los usuarios.

En este entorno de desarrollo reside una versión de pruebas del Back-End de Neki. De esta forma no se compromete el correcto funcionamiento de la plataforma.

Todas las pruebas han sido realizadas de manera conjunta entre el autor del TFG y Neki. El autor se ha encargado de que el middleware pase las diferentes pruebas mientras que Neki se ha centrado en que las pase su Back-End.

Las pruebas realizadas han sido:

- Comprobar el correcto funcionamiento del middleware en el servidor de Neki.
- Probar que la comunicación Middleware-Neki se realiza de manera correcta.
- Verificar la correcta interacción entre el cliente y el sistema.
- Que el middleware es capaz de procesar las peticiones del Back-End y viceversa.
- Comprobar que el usuario final es capaz de recibir las notificaciones generadas por el sistema cuando corresponde.

c. Validación de los requisitos

Una vez completada la implementación de la aplicación, es preciso realizar un análisis del funcionamiento del sistema respecto a los requisitos establecidos, lo que refuerza las decisiones de diseño e implementación tomadas a lo largo del desarrollo del proyecto.

Para ello, se toman en consideración todos los requisitos definidos en el apartado "**Requisitos**" del presente documento, y se verificara que se cumple dichos requisitos a través del uso de la aplicación.

La tabla que se muestra a continuación sirve como plantilla para la validación de los requisitos respecto a la implementación:

Requisito	
Descripción	
Prueba realizada	
Validación	

Figura 32. Plantilla evaluación requisitos

A continuación, se incluye una breve descripción de los campos que componen la tabla:

- **Identificador**: Este campo corresponde al identificador del requisito. Dicho identificador será único e inconfundible, de tal forma que cada requisito sea identificable sin posibilidad de error.
  Cada uno de los identificadores seguirá la siguiente nomenclatura:
  - Requisitos funcionales: RF- y un número entre 01 y 99.
- **Descripción**: Campo que incluye una descripción del requisito en cuestión.
- **Prueba**: Describe las acciones a realizar para comprobar el requisito.
- **Validación:** Determina si, con las pruebas realizadas, se puede afirmar que el requisito en cuestión se cumple durante la ejecución de la aplicación.

REQUISITOS FUNCIONALES

Requisito	RF-01
Descripción	Para poder notificar a los tutores de que algo puede ir mal dentro de la vivienda Neki quiere recibir todas actividades de la vivienda en su plataforma.
Prueba realizada	Para validar este requisito se ha puesto en funcionamiento el sistema colocando un Beacon en una puerta. En el momento que se produce una actividad de movimiento se ha comprobado si en el registro de actividades se ha almacenado este evento. En caso afirmativo se da por validado el requisito.
Validación	Correcta

Figura 33. Validación RF-01

Requisito	RF-02
Descripción	Este sistema se basa en establecer unas reglas de inferencia que corresponden a actividades cotidianas. El tutor es quien mejor conocer las rutinas del monitorizado, por esta razón se quiere poder establecer las reglas de inferencia a través de la APP móvil.
Prueba realizada	Para validar este requisito se ha credo una regla de inferencia, mover la puerta cada minuto, a través de una cuenta de usuario. La prueba es válida si la regla se ha registrado en el sistema.
Validación	Se ha validado por el alumno pero no se ha validado en tutores reales.

Figura 34. Validación RF-02

Requisito	RF-03
Descripción	El tutor es quien está al cargo del monitorizado. Quiere saber en todo momento que todo va bien en la vivienda del monitorizado y ser avisado cuando algo no vaya bien para tomar las acciones oportunas.
Prueba realizada	A partir de la prueba anterior, se ha estado un minuto sin mover una puerta. El usuario ha tenido que recibir una notificación informando sobre esta actividad. En caso afirmativo se da por validado el requisito.
Validación	Se ha validado por el alumno pero no se ha validado en tutores reales.

Figura 35. Validación RF-03

Requisito	RF-04
Descripción	Neki y el tutor quieren conocer el porcentaje de batería de lo Beacons para conocer cuando es necesario cambiar la batería.
Prueba realizada	Para validar este requisito se ha puesto en funcionamiento el sistema. La prueba es válida si el middleware es capaz de recibir y enviar mensajes que contienen información del estado de la batería de los Beacons.
Validación	Correcta

Figura 36. Validación RF-04

Requisito	RF-05
Descripción	Neki quiere detectar los Beacons que han dejado de emitir durante un tiempo determinado para poder solucionar el problema lo antes posible.
Prueba realizada	Para validar este requisito se ha puesto en funcionamiento el sistema. La prueba ha consistido en conocer que sensores han dejado de emitir durante un minuto. Al desactivar un Beacon durante un minuto el middleware ha tenido que ser capaz de detectar la inactividad del Beacon. La prueba se da por válida si se ha generado un mensaje de inactividad por parte del Beacon desconectado.
Validación	Correcta

Figura 37. Validación RF-05

Requisito	RF-06
Descripción	Una persona no autorizada se puede conectarse al servicio y hacer uso de él. Para evitar esta situación Neki quiere que se procesen solo aquellas tramas de gateways que están registrados en el sistema.
Prueba realizada	Para validar este requisito se han borrado los registros de un gateway. La prueba es válida si el middleware recibe las tramas de este gateway, pero no las procesa.
Validación	Correcta

Figura 38. Validación RF-06

Requisito	RF-07
Descripción	Neki va a ofrecer este producto como un servicio de alquiler. Cuando una persona se quiera dar de baja al servicio se quiere poder desactivar su gateway para no proporcionarle servicio. A su vez, cuando una persona comience a usar este servicio se quiere dar de alta su gateway.
Prueba realizada	Para validar este requisito se han registrado un gateway como desactivado. La prueba es válida si el middleware recibe las tramas de este gateway, pero no las procesa hasta que no se activa el gateway. En ese preciso momento el middleware debe procesar las tramas de este gateway.
Validación	Correcta

Figura 39. Validación RF-07

Requisito	RF-08
Descripción	Para tener un feedback del sistema y un control sobre las actividades en las viviendas Neki quiere tener un registro con actividades producidas.
Prueba realizada	Para validar este requisito se han generado los diferentes tipos de actividades (movimiento, batería, inactividad…) y se ha comprobado que se almacenan correctamente en la base de datos.
Validación	Correcta

Figura 40. Validación RF-08

REQUISITOS NO FUNCIONALES

Requisito	RNF-01
Descripción	Debido a que la plataforma de Neki está proporcionando actualmente servicio se quiere que la integración de este nuevo sistema a su plataforma se haga sin modificar la plataforma actual.
Prueba realizada	No se han realizado pruebas para este requisito. Se ha desarrollado el middleware con las mismas tecnologías que la plataforma de Neki y pensando en su protocolo de comunicación. No se ha realizado ninguna modificación importante en su plataforma.
Validación	Correcta

Figura 41. Validación RNF-01

Requisito	RNF-02
Descripción	Debido a su fácil instalación y larga autonomía Neki quiere usar sensores acelerómetro para detectar los movimientos de una puerta.
Prueba realizada	Se ha realizado un estudio de los Beacon en el apartado de evaluación para conocer la validez de este requisito.
Validación	Correcta

Figura 42. Validación RNF-02

Requisito	RNF-03
Descripción	Neki quiere estimar el tráfico de red móvil que se produce por el gateway y así tener una idea del coste de este servicio.
Prueba realizada	Se ha realizado un análisis de consumo en el apartado de evaluación. Con esto se da por validado el requisito.
Validación	Correcta

Figura 43. Validación RNF-03

Requisito	RNF-04
Descripción	Saber si se ha abierto una puerta mediante un sensor es mínimamente invasivo, mientras que el uso de una cámara es invasivo para el monitorizado. Este sistema quiere respetar lo máximo posible la intimidad y privacidad del monitorizado.
Prueba realizada	El uso únicamente de sensores Beacons garantizan que el sistema de monitorización es mínimamente intrusivo.
Validación	Correcta

Figura 44. Validación RNF-04

9. Conclusiones

Se ha creado un sistema-servicio que, con el uso de unos Beacons y un gateway, permite monitorizar a una persona de forma no intrusiva, avisando al tutor cuando algo va mal en la vivienda.

Para instalar este servicio en una vivienda únicamente se necesitan unos Beacons y un gateway. Su Instalación en una vivienda consiste simplemente en colocarlo los Beacons en las puertas deseadas y conectar el gateway a la red eléctrica.

El problema de este servicio es que hay que preconfigurar estos dispositivos. No disponer de un firmware propio o modificado conlleva configurarlos uno a uno.

Debido a sus características, Neki solo ha estado interesado en usar Beacons con acelerómetro para detectar los movimientos de una puerta.

Este requisito ha llevado a cabo la necesidad de evaluar su funcionamiento. La evaluación ha justificado que estos Beacons son capaces de detectar los movimientos de las puertas.

Otro punto de interés de estos Beacons era conocer si la emisión de tramas por umbral era válida para la detección de los movimientos. La evaluación ha determinado que el umbral establecido no es lo suficientemente sensible.

Si el fabricante permitiera configurar este umbral sería posible conocer qué valor de sensibilidad detecta los movimientos. Usar este modo de funcionamiento implicaría un menor consumo de energía en los Beacons. Además, al emitir tramas únicamente cuando se produce un movimiento haría que no fuese necesario filtrar la señal, las tramas emitidas corresponderían a movimientos de puerta.

No se ha estudiado el consumo de energía de los Beacons debido a que durante el desarrollo de este trabajo, más de tres meses, no se ha producido un descenso en sus porcentajes. Lo que presupone que van a tener una larga duración.

MQTT ha permitido que el sistema sea muy escalable, además de generar muy poco tráfico. En la evaluación de los Beacons se ha conectado un nodemcu esp8266 al middleware sin realizar modificaciones.

A todo esto, el gateway va a utilizar de una red móvil que cobra por consumo. Al ser MQTT un protocolo de bajo tráfico, añadir un Beacon al sistema supone una previsión de consumo adicional de 20 MB al mes. Un consumo realmente bajo.

Con respecto al desarrollo del middleware, el uso de Node.JS y librerías de uso libre ha permitido que con poco código se tenga una versión completamente funcional. A todo esto, cabe mencionar que se puede balancear la carga de trabajo virtualizando y replicando este middleware.

La integración del middleware con la plataforma de Neki ha sido una tarea más sencilla de lo esperada. Al ser dos componentes que usan las mismas tecnologías no se han producido incompatibilidades y no ha hecho falta el uso de ninguna librería auxiliar para comunicar estos componentes.

Por último, con respecto al back-end y la interacción con los clientes no se ha proporcionado ningún tipo de información por parte de Neki. Por lo tanto, no se pueden sacar conclusiones de esta parte del sistema.

Tras concluir el desarrollo del sistema y realizar una serie de pruebas, se puede decir que el sistema cumple con los requisitos establecidos. Todo esto permite afirmar que se ha conseguido cumplir el objetivo principal que se establecieron en un principio.

10. Gestión del proyecto

A continuación, se detalla la metodología seguida para llevar a cabo este proyecto, así como la duración de el mismo.

a. Metodología

La metodología se ha basado en la realización de reuniones con Neki. En estas reuniones se han ido comentado los avances del proyecto y, a partir de estos avances, se ha decidido que tareas se iban a llevar a cabo hasta la próxima reunión.

Debido a que el desarrollo de la aplicación se ha llevado por cuenta propia, junto con reuniones cada 15 días aproximadamente, se calculó que el trabajo estaría terminado para finales de Julio.

Al comienzo de este proyecto, el 27 de marzo de 2018, se llevó a cabo una reunión inicial por parte del director, ponente y el autor. En esta primera reunión se comentó la idea general del proyecto y sus objetivos. Por último, se decidió que se iba a adquirir un kit de desarrollo para aplicaciones IoT.

Tras esta reunión se realizó la compra de este kit el cual incluye los Beacons y el gateway mencionados en este proyecto. Con esta compra el fabricante envio la documentación del kit. En esta documentación se habla de las características de los dispositivos, su funcionamiento y que es lo que se puede configurar.

El trabajo se ha dividido en una serie fases diferenciadas, las cuales entre fase y fase se ha llevado a cabo una reunión:

Análisis del kit de desarrollo. Para analizar este kit de desarrollo se ha estudiado toda la documentación proporcionada por el fabricante. Este primer análisis ha permitido saber cómo funciona este kit y que posibilidades ofrece.

Posteriormente, se llevó a cabo una reunión para hablar sobre el kit de desarrollo y se decidió que había que estudiar la viabilidad de los Beacons para este sistema.

Estudio de los Beacons. Después de analizar el kit de desarrollo se ha realizado el estudio de los Beacons con el objetivo de saber si son capaces de detectar los movimientos de una puerta. Este estudio ha llegado a la conclusión de que estos dispositivos son viables para el desarrollo del sistema.

Una vez validad su viabilidad, se produjo una reunión con Neki para comentar los resultados del estudio y se decidió que era el momento de implementar el middleware y probarlo.

Diseño e implementación del middleware. Una vez comprobada la validez de los Beacons se ha diseñado e implementar el middleware. La decisión de crear un middleware fue tomada en las fases iniciales del proyecto, pero no se ha implementado hasta conocer la fiabilidad de los Beacons. En esta fase es cuando se decide usar MQTT como protocolo de comunicación frente a HTTP.

Pruebas sobre el middleware. Para probar el correcto funcionamiento del middleware se ejecutó el sistema en una vivienda, en estas pruebas se comprobó que el sistema de monitorización es capaz de detectar las actividades de movimiento, informar del estado de la batería y detectar sensores caídos.

Una vez probada su validez se realizó una reunión para determinar cómo iba a realizarse la comunicación middleware Back-end.

Comunicación entre el middleware y el Back-End. Se habló con Neki como y donde se iba a alojar el middleware creado. Además, se determinó con sus ingenieros un protocolo de interacción entre el middleware y el Back-End.

Adaptación del middleware. Posteriormente se reestructuro el middleware y se adaptó para que soportara el protocolo de mensajes pactado con Neki. Una vez modificado, se creó un repositorio privado para el trabajo en paralelo con Neki.

Durante las siguientes fases de integración el autor del TFG y Neki han trabajado conjuntamente para llevar a cabo la correcta integración del middleware con la plataforma de Neki.

Adaptación del Back-End. A la vez que se ha modificado el middleware se ha adaptado el Back-End por parte de Neki. Debido a que se quiere mantener su confidencialidad la confidencialidad de este servidor en este trabajo no se entra en detalles sobre este componente. Pero si es interesante mencionar que es una fase del proyecto final.

Integración del middleware en la plataforma de Neki. En esta última fase se ha integrado el middleware en un servidor de desarrollo. Se han realizado una serie de pruebas de manera conjunta para comprobar el correcto funcionamiento de la aplicación. El autor del trabajo final de grado se ha encargado de corregir los fallos por parte del middleware mientras que Neki ha corregido los fallos por parte del Back-End.

Memoria. Por último, se ha redactado la documentación de este proyecto. El resultado será este presente documento en el que se describirá el trabajo realizado a lo largo del desarrollo del proyecto evaluando los resultados obtenidos.

b. Planificación final

El proyecto se ha llevado a cabo según las fechas estimadas realizando un trabajo de unas 470 horas aproximadamente.

A todo este tiempo hay que sumar las reuniones llevadas a cabo unas reuniones con la empresa Neki. Estas reuniones se han llevado a cabo cada 15 días durante las fases iniciales del proyecto. Las reuniones han estado formadas por el ponente, tres empleados de Neki (uno de ellos el director ejecutivo) y el autor del TFG, cada una han tenido una duración de una hora.

Se ha retrasado el desarrollo del proyecto según lo planeado debido a unos diversos motivos. En primer lugar, el mes de junio coincidió con época de exámenes, por lo que se decremento el rendimiento este mes.

En segundo lugar, el parón que aparece la semana del 9 al 15 de agosto se debe a una salida vacacional del alumno.

Por último, ponerse de acuerdo sobre las fechas de realización del proyecto ha supuesto unos pequeños parones.

El siguiente diagrama de Gantt muestra la planificación final de tiempo invertido durante el desarrollo de este trabajo de fin de grado. Se puede observar como hay periodos en los que no se ha realizado ninguna actividad.

Estos periodos corresponden a la organización de las reuniones y a la primera semana de vacaciones de agosto por parte del alumno.

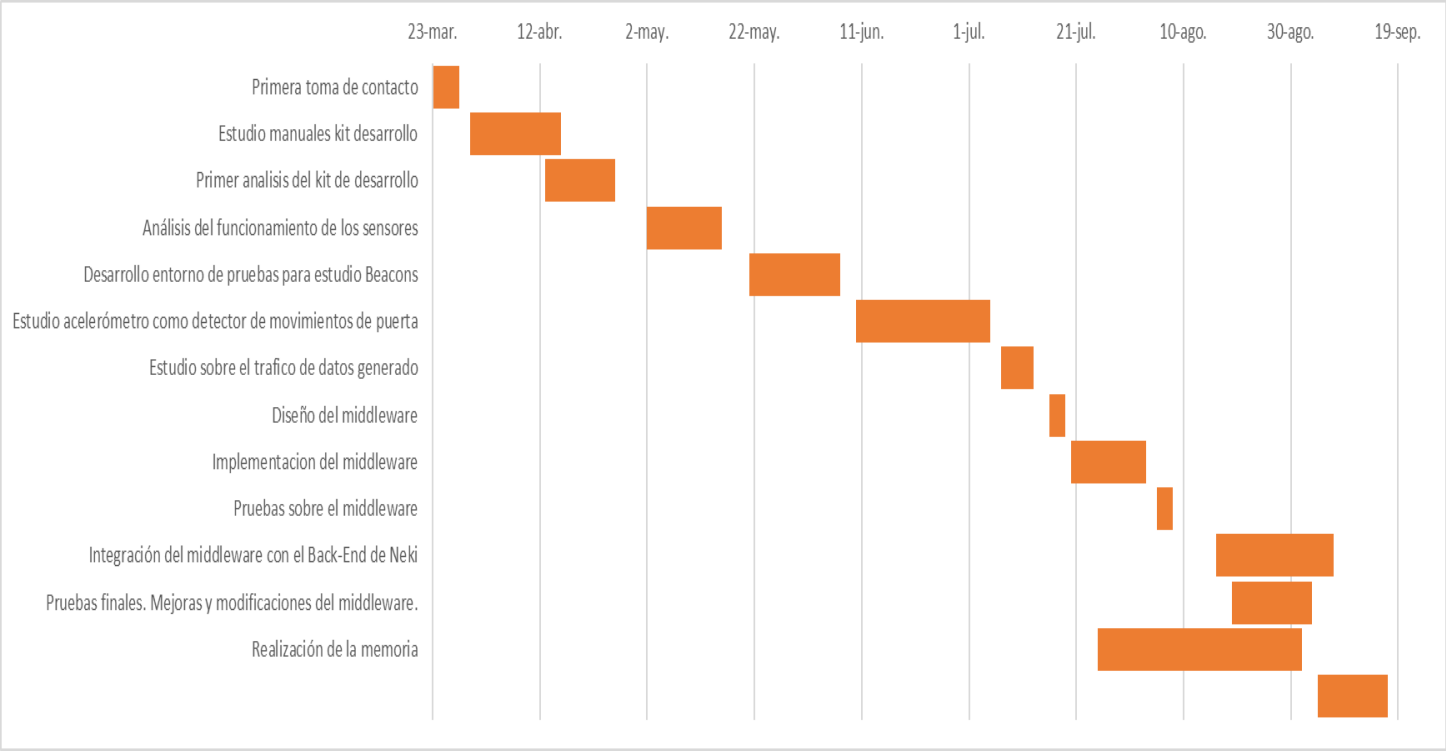

Figura 45. Diagrama de Gantt

Sobre las horas invertidas en el proyecto, será un dato aproximado ya que no se ha llevado una cuenta exhaustiva. Contando que me quedaban asignaturas por aprobar en junio habré invertido unas 2,5 horas de media al día contando semanas y fines de semana.

Como quiero valorar económicamente el esfuerzo invertido en el proyecto, he considerado interesante crear un pequeño presupuesto que permita conocer los gastos asociados al desarrollo.

Para mantener la confidenciad de los dispositivos proporcionados por Neki, no se va a tener en cuenta el coste de estos dispositivos usados para el proyecto.

El coste de los recursos humanos para el proyecto va a incluir únicamente al autor, con respecto a Neki y el ponente no se ha realizado ningún tipo de estimación, sería interesante considerarlo, pero quiero valorar mi esfuerzo.

El coste estimado para el autor, considerado como programador con baja experiencia a efectos prácticos, será de 6.25€/hora.

En la siguiente tabla se muestran las horas invertidas para cada una de las fases del proyecto, así como el total de horas invertidas:

Fase	Horas Invertidas	Coste (euros)
Primera toma de contacto	12,5	77,5
Análisis y aprendizaje	110	682
Estudio	120	744
Diseño Middleware	7,5	46,5
Implementación I	35	217
Evaluación I	8	49,6
Implementación II	33	204,6
Evaluación II	42	260,4
Memoria	110	682
Costes finales	478	2963,6

Figura 46. Estimación de costes humanos y económicos.

En esta tabla cabe destacar la diferencia de horas entre la fase de análisis y aprendizaje y la de implementación del middleware. Como el fabricante ha proporcionado una documentación para aprender a usar su kit se ha tenido que realizar un aprendizaje exhaustivo de sus manuales.

A todo esto, hay que sumar que al ser la primera vez que desarrollo un proyecto usando Node.JS he tenido que adquirir conocimientos previos. Por último, se ha tenido que buscar una forma de extraer y representar correctamente los datos de los Beacons emitidos en hexadecimal.

En la fase de implementación I se he considerado solo la implementación del middleware, no el aprendizaje de Node.JS ni la forma de extraer los datos de los Beacons. Esta es la razón por la cual hay tanta diferencia de tiempo entre estas fases.

Gracias a este análisis y aprendizaje, desarrollar una futura aplicación en Node.JS va a suponer un menor tiempo.

Por último, la siguiente tabla recoge la duración de cada una de las tareas.

Tarea	Fecha de inicio	Fecha de finalización	Duración (días)	Duración (Horas)
Primera toma de contacto	23-mar.	28-mar.	5	12,5
Análisis y aprendizaje				
Estudio manuales kit desarrollo	30-mar.	16-abr.	17	42,5
Primer análisis del kit de desarrollo	13-abr.	26-abr.	13	32,5
Análisis del funcionamiento de los sensores	2-may.	16-may.	14	35
Estudio				
Desarrollo entorno de pruebas para estudio Beacons	21-may.	7-jun.	17	42,5
Estudio acelerómetro como detector de movimientos de puerta	10-jun.	5-jul.	25	62,5
Estudio sobre el tráfico de datos generado	7-jul.	13-jul.	6	15
Diseño				
Diseño del middleware	16-jul.	19-jul.	3	7,5
Implementación y evaluación I				
Implementación del middleware	20-jul.	3-ago.	14	35
Pruebas sobre el middleware	5-ago.	8-ago.	3	7,5
Implementación y evaluación II				
Integración del middleware con el Back-End de Neki	16-ago.	7-sep.	22	55
Pruebas finales. Mejoras y modificaciones del middleware.	19-ago.	3-sep.	15	37,5
Memoria				
Realización de la memoria	25-jul.	1-sep.	38	95
Revisión de la memoria	4-sep.	17-sep.	13	32,5

Figura 47. Duración de las tareas del proyecto.